Applications of Computational Topology to the Visualization of Scalar Fields

Vom Fachbereich Informatik der Technischen Universität Kaiserslautern

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Natallia Kotava

Datum der wissenschaftlichen Aussprache: 16. Mai 2014

Dekan:	Prof. Dr. Klaus Schneider
Vorsitzende der Prüfungskommission:	Prof. Dr. Katharina Zweig
Erster Berichterstatter:	Prof. Dr. Christoph Garth
Zweiter Berichterstatter:	Prof. Dr. Thomas Wischgoll

D 386

Acknowledgements

I want to thank all the people and organizations who supported me during the work on this dissertation.

First of all, I wold like to thank my advisors and referents of this work Prof. Dr. Christoph Garth, Prof. Dr. Hans Hagen and Prof. Dr. Thomas Wischgoll. I want to thank Prof. Dr. Hans Hagen for the opportunity to work on my dissertation in an international environment, being a part of the International Research Training Group "Visualization of Large and Unstructured Data Sets - Applications in Geospatial Planning, Modeling, and Engineering" (IRTG 1131) and the freedom he gave me during my research work. I am thankful to Prof. Dr. Christoph Garth for interesting scientific discussions, motivation and support during the end phase of my PhD.

Secondly, I want to thank my collaboration partners without whom it would not be possible to develop all the ideas for this thesis. I am thankful to Prof. Dr. Charles Hansen for a warm welcome to Salt Lake City and Scientific Computing Institute, for fruitful thematic discussions, engagemount and interest at my work. I would like to thank Dr. Attila Guylassy and Prof. Dr. Valerio Pascussi for introducing Computational Topology to me and for friendly cooperation during my stay in Utah. Special thanks to Dr. David Günther and Dr. Jan Reininghaus for structured and productive work together. I want to thank Dr. Aaron Knoll for many brilliant ideas, his open-mindness and help.

Also I would like to thank my friends and colleagues for many scientific discussions and personal support: Dr. Fang Chen, Dr. Christian Wagner, Rolf Westerteiger, Dr. Sebastian Thelen, Alexander Petry, Alexander Neundorf, Christopher Armbrust. Last, but not least, special thanks to my family and people who were close to me during the period of research for their acceptance and support.

Abstract

This thesis discusses several applications of computational topology to the visualization of scalar fields. Scalar field data come from different measurements and simulations. The intrinsic properties of this kind of data, which make the visualization of it to a complicated task, are the large size and presence of noise. Computational topology is a powerful tool for automatic feature extraction, which allows the user to interpret the information contained in the dataset in a more efficient way. Utilizing it one can make the main purpose of scientific visualization, namely extracting knowledge from data, a more convenient task.

Volume rendering is a class of methods designed for realistic visual representation of 3D scalar fields. It is used in a wide range of applications with different data size, noise rate and requirements on interactivity and flexibility. At the moment there is no known technique which can meet the needs of every application domain, therefore development of methods solving specific problems is required. One of such algorithms, designed for rendering of noisy data with high frequencies is presented in the first part of this thesis. The method works with multidimensional transfer functions and is especially suited for functions exhibiting sharp features. Compared with known methods the presented algorithm achieves better visual quality with a faster performance in presence of mentioned features. An improvement on the method utilizing a topological theory, Morse theory, and a topological construct, Morse-Smale complex, is also presented in this part of the thesis. The improvement allows for performance speedup at a little precomputation and memory cost.

The usage of topological methods for feature extraction on a real world dataset often results in a very large feature space which easily leads to information overflow. Topology simplification is designed to reduce the number of features and allow a domain expert to concentrate on the most important ones. In the terms of Morse theory features are represented by critical points. An importance measure which is usually used for removing critical points is called homological persistence. Critical points are cancelled pairwise according to their homological persistence value. In the presence of outlier-like noise homological persistence has a clear drawback: the outliers get a high importance value

assigned and therefore are not being removed. In the second part of this thesis a new importance measure is presented which is especially suited for data with outliers. This importance measure is called scale space persistence. The algorithm for the computation of this measure is based on the scale space theory known from the area of computer vision. The development of a critical point in scale space gives information about its spacial extent, therefore outliers can be distinguished from other critical points. The usage of the presented importance measure is demonstrated on a real world application, crater identification on a surface of Mars.

The third part of this work presents a system for general interactive topology analysis and exploration. The development of such a system is motivated by the fact that topological methods are often considered to be complicated and hard to understand, because application of topology for visualization requires deep understanding of the mathematical background behind it. A domain expert exploring the data using topology for feature extraction needs an intuitive way to manipulate the exploration process. The presented system is based on an intuitive notion of a scene graph, where the user can choose and place the component blocks to achieve an individual result. This way the domain expert can extract more knowledge from given data independent on the application domain. The tool gives the possibility for calculation and simplification of the underlying topological structure, Morse-Smale complex, and also the visualization of parts of it. The system also includes a simple generic query language to acquire different structures of the topological structure at different levels of hierarchy.

The fourth part of this dissertation is concentrated on an application of computational geometry for quality assessment of a triangulated surface. Quality assessment of a triangulation is called surface interrogation and is aimed for revealing intrinsic irregularities of a surface. Curvature and continuity are the properties required to design a visually pleasing geometric object. For example, a surface of a manufactured body usually should be convex without bumps of wiggles. Conventional rendering methods hide the regions of interest because of smoothing or interpolation. Two new methods which are presented here: curvature estimation using local fitting with Bézier patches and computation of reflection lines for visual representation of continuity, are specially designed for assessment problems. The examples and comparisons presented in this part of the thesis prove the benefits of the introduced algorithms. The methods are also well suited for concurrent vi-

sualization of the results from simulation and surface interrogation to reveal the possible intrinsic relationship between them.

Zusammenfassung

Diese Arbeit untersucht verschiedene Anwendungen von Algorithmischer Topologie zur Visualisierung von Skalarfelddaten. Skalarfelddaten entstehen bei verschiedenen physikalischen Messmethoden oder durch Simulation. Sie sind häufig sehr groß und mit Rauschen behaftet, was eine besondere Herausforderung für deren Visualisierung ist. Algorithmische Topologie ist ein leistungsfähiges Werkzeug zur automatischen Merkmalsextraktion, die es einem Nutzer erlaubt die in den Daten enthaltenen Informationen effizienter und bequemer zu extrahieren und zu interpretieren.

Volume Rendering beschreibt eine Klasse von Methoden die für die realistische visuelle Representation von dreidimensionalen Skalarfeldern entwickelt wurden. Sie werden in einem breiten Anwendungsspektrum mit unterschiedlichen Anforderungen an Datenmengen, Rauschen, Interaktivität und Flexibilität eingesetzt. Bisher ist keine Methode bekannt, die die Bedürfnisse sämtlicher Anwendunsbereiche abdecken kann. Daher ist häufig die Auswahl bzw. Entwicklung spezifischer Methoden notwendig. Im ersten Teil dieser Arbeit wird eine Methode vorgestellt, die besonders für die Visualisierung von Daten geeignet ist, die stark mit hochfrequentem Rauschen behaftet sind. Diese Methode kann mit mehrdimensionalen Transferfunktionen umgehen und ist besonders geeignet für Transferfunktionen die sehr scharf ausgeprägte Funktionsverläufe enthalten. Im Vergleich zu bekannten Methoden bietet der vorgestellte Algorithmus unter diesen Voraussetzungen bessere visuelle Qualität und gleichzeitig höhere Effizienz. Um die Methode weiter zu verbessern wird außerdem gezeigt, wie die Morse-Theorie, eine Teilgebiet der Topologie, und ihre Konstrukte angewandt werden können, so dass die Berechnungsgeschwindigkeit bei lediglich geringfügigen Aufwand und Speicherverbrauch zur Vorberechung gesteigert werden kann.

Die Anwendung von topologischen Methoden zur Merkmalsextraktion führt bei realen Datensätzen häufig zu einem sehr großen Merkmalsraum, der den Benutzer leicht überlasten kann. Methoden zur topologische Vereinfachung dienen daher dazu die Anzahl der Merkmale zu reduzieren und es dem Experten zu ermöglichen, sich auf die relevanten Merkmale zu konzentrieren. In der Morse-Theorie werden Merkmale durch sogenannte kritische

Punkte repräsentiert. Eine Möglichkeit die Anzahl der kritischen Punkte zu begrenzen, ist ein Wichtigkeitsmaß wie die homologische Persistenz zu verwenden. Kritische Punkte heben sich paarweise gegenseitig auf, entsprechend ihrem homologischen Persistenzwerts. Jedoch hat die homologische Persistenz bei Ausreißern in den Daten einen entscheidenden Nachteil: Die Ausreißer erhalten einen besonders hohen Wichtigkeitswert und werden daher nicht entfernt. Im zweiten Teil der Arbeit wird daher ein neues Wichtigkeitsmaß "Scale Space Persistence" eingeführt, das besonders für Daten mit Ausreißern geeignet ist. Der Algorithmus zur Berechnung dieses Maßes basiert auf der Skalenraumtheorie, die z.B. in der Bildverarbeitung häufig verwendet wird. Die Entwickung eines kritischen Punkts im Skalenraum gibt Aufschluss über deren räumliche Ausdehnung, so können Ausreißer leicht von anderen kritischen Punkten unterschieden werden. Das vorgestellte Wichtigkeitsmaß wird anhand einer realen Anwendung, der Krateridentifikation auf der Marsoberfläche, demonstriert.

Der dritte Teil der Arbeit präsentiert ein System zur generischen interaktiven topologischen Analyse und Untersuchung. Die Entwicklung eines solchen Systems ist motiviert durch die Tatsache, dass topologische Methoden als zu kompliziert und schwer zu verstehen wahrgenommen werden, da die Anwendung von Topologie zur Visualisierung ein tieferes Verständnis des mathematischen Hintergrunds erfordern. Ein Domänenexperte der einen bestimmten Datensatz durch toplogische Methoden zur Merkmalsextraktion untersuchen möchte benötigt daher ein intuitives Werkzeug, das ihm erlaubt den Explorationsprozess zu steuern. Das vorgestellte System basiert auf einer anschaulichen Darstellung durch einen Szenengraphen, den der Benutzer durch Hinzufügen und Anordnen von Bausteinen an seine Bedürfnisse anpassen kann. Auf diese Art kann der Domänenexperte zusätzliche Informationen aus einem Datensatz extrahieren und eine anwendungsspezifische Verarbeitung basierend auf generischen Methoden zusammenstellen. Das Werkzeug erlaubt es die topologische Struktur, den sogenannten Morse-Smale Complex, der Daten zu berechnen, zu vereinfachen und Teile davon zu visualisieren. Das System beinhaltet eine einfache Anfragesprache um unterschiedliche Teile der Struktur auf verschiedene Hierarchieebenen abzufragen.

Der vierte Teil der Arbeit konzentriert sich auf Anwendungen von Algorithmischer Geometrie zur Qualitätsbewertung einer triangulierten Oberfläche. Diese auch "Surface Interrogation" genannte Methode ist darauf ausgelegt Unregelmäßigkeiten einer Oberfläche

aufzudecken. Krümmung und Kontinuität sind Faktoren, die die Ästhetik eines Objekt maßgeblich beeinflussen, z.B. bei der Gestaltung eines Produktgehäuses. Herkömmliche Visualisierungsmethoden verstecken Unregelmäßigkeiten durch Glättung oder Interpolation. Zwei neue Methoden werden vorgestellt, mit deren Hilfe Kontinuität besser untersucht werden kann: ”Curvature Estimation Using Local Fitting with Bézier Patches”, sowie ”Computation of Reflection Lines”. Beispiele und Vergleiche mit bestehenden Methoden zeigen den Vorteil der vorgestellten Algorithmen. Die Methoden sind ebenso sehr gut geeignet um Simulationsergebnisse, z.B. zur Aerodynamik eines Objekts, mit den Oberflächeneigenschaften zu vergleichen und mögliche Zusammenhänge aufzudecken.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Scientific visualization is used for visual representation of the data collected using various measurements or simulations. Scalar field visualization deals with the uni- or multivariate scalar data given on some (usually structured) grid. A simple example of such data is a CT scan of a human brain. Traditionally the domain expert exploring the data has a specific question that should be answered using the visualization. If this question is known beforehand, the visualization and its parameters can be specifically designed for this application domain. However by predetermining these parameters the user is limited to the specific aspect of the data and is not able to extract additional information potentially present in the data. Therefore one of the goals of modern scientific visualization is to provide interactivity and flexibility while exploring the data. One of the main challenges in this area is to improve on the performance of the visualization method to allow for an interactive exploration of the data, even if the dimensions of the dataset are very large. Another challenge is to give the user proper tools to interact with the data and extract the information he is interested in.

One of the most common methods to visualize volumetric scalar data is volume rendering. Typically the data value at each grid point is mapped to the optical properties, color and opacity, which can then be visually represented. The mapping is determined by a transfer function. Through adjusting this function the user can influence the visual result, high-

lighting the objects of interest and hiding the areas he is not interested in. For example, on a CT scan of a tooth one can be interested in seeing only the dentin. The value range of dentin is known, so one can design a transfer function, which highlights only this value range.

One of the issues of volume rendering is performance. The process of volume rendering is the computation of a discrete equivalent of the volume rendering integral for each screen pixel. The size of the volume, the topology of the transfer function and the size of the viewport often make it impossible to compute the integral at an interactive frame rate. Another issue is transfer function design. Although some of the methods for automatic transfer function design exist in the literature, it is still a common practice to manually choose the transfer function and optimize it in an iterative way based on the visual feedback. The reason for this is that the transfer functions are heavily dependent on the specific task. It is hard to choose a good transfer function for some kinds of data, for example, if the value ranges of different objects intersect. In this case one needs to take into account additional information present in the data, for example, the gradient magnitude value, which motivates the use of multidimensional transfer functions. Performance optimization can be achieved by reducing the size of the data, precomputing some intermediate results or by exploiting the characteristics of the data. If the transfer function has to be chosen interactively, precomputation is usually not an option. Therefore methods that provide a good tradeoff between quality and interactivity are required.

If large and complex datasets are visualized the user is easily overwhelmed. One way to avoid information overflow is extracting and visualizing specific features instead of visualizing unprocessed data. A feature can be for example objects of specific shape or areas of extreme values. A powerful approach for feature extraction is to utilize the mathematical theory of differential topology. The combinatorial equivalent of this theory and computational methods have been developed in the last decade (the details will be described later in chapter 3). Due to the expressiveness of differential topology it can be applied to a large variety of applications. However their complex nature makes them hard to understand and interpret for a domain expert. Therefore, interactive tools are required that support the user in applying topology-based methods and adjusting the respective parameters.

Even when using topology-based feature extraction methods, the amount of features extracted from a real-world dataset can be too large for a user to be able to visually investigate them. Therefore he needs support to distinguish between important and unimportant features. Importance measures are needed to be able to filter out unimportant features, for example, introduced by noise. This thesis addresses the mentioned issues in scalar field visualization with respect to volume rendering and topology-based feature extraction.

1.2 Overview and contribution

This thesis presents several methods for the application of computational topology to the visualization of scalar fields. In particular the two main areas are covered: volume rendering (Chapter 4) and topology-based feature extraction (Chapters 5 and 6). Additionally, chapter 7 presents an application of computational geometry.

The author of this dissertation was a part of the International Research Training Group 1131 (IRTG) financed by the German Research Foundation (Deutsche Forschungsgesellschaft (DFG)). The international graduate school supported diverse national and international collaborations, therefore most of the publications on the base of which this thesis is built resulted from the work together with the partner institutions.

Chapter 2 describes the theoretical background required for better understanding of the terminology and the notation throughout this thesis. Chapter 3 summarizes the state of the art relevant to the contributions of this work. There are brief descriptions of the state of the art in other areas when necessary in the corresponding chapters.

Chapter 4 discusses a method for volume rendering with multidimensional transfer functions, multidimensional peak finding. The method identifies peaks along the image of the ray in the transfer function space. This approach is useful for specifying transfer functions with greater precision and for accurately rendering noisy volume data at lower sampling rate.

Section 4.1 gives a review of the one-dimensional predecessor of the presented method, the unidimensional peak finding technique. Section 4.2 describes the algorithm itself: dealing with separable and non-separable transfer functions, the different ways of sampling in the transfer function space and ray parameterization, implementation issues of the

method. Section 4.3 presents some examples for application of the method and compares it with the compelling methods. Section 4.4 proposes an optimization of the technique using the topology-based transfer function domain subdivision, which helps to increase the volume rendering performance.

The contribution of this chapter is the novel method for volume rendering with multi-dimensional transfer functions, which gives qualitatively better results than the methods known before. The algorithm is especially efficient in the case of a transfer function with steep local extrema. An example of such a function in one dimension is Kronecker delta function, which is zero everywhere except of one point, where it takes the maximal possible value of one:

$$\delta[n] = \begin{cases} 0, & n \neq a \\ 1, & n = a, \end{cases}$$

where a is some scalar value. Although Kronecker delta is an extreme example of the steepest slope, it represents the ideal transfer function choice, when isosurface rendering is desired. To perform direct volume rendering with such a transfer function, the sampling rate along the ray has to be chosen very high, resulting in a performance slowdown. Peak finding reduces the requirement on the sampling rate, adaptively putting additional samples where needed. Because of a lower requirement on the initial sampling rate *of the data*, the peak finding method also effectively deals with noisy data, producing qualitatively compelling results.

The contents of this chapter were published in [KKS+12] and [KKH13]. The methods presented resulted from a collaborative work with multiple people, mainly with Dr. Aaron Knoll, who was a part of IRTG 1131 back then. Dr. Knoll is an author of the idea of one-dimensional peak finding method, which was used as an inspiration for the multidimensional algorithm. Although the name ”peak finding” was kept, the reader will see later in Chapter 4 that the extension is not straightforward and requires a different methodology. The author of this thesis developed the main ideas, made research on the state of the art, participated in the implementation and prepared both publications mentioned earlier.

Chapter 5 introduces a novel importance measure for the visualization of critical points in 2D scalar fields. This measure is based on a combination of the deep structure of the scale space with the concept of homological persistence. Section 5.1 briefly discusses the

theory and state of the art in the area of scale space and gives a review of the method for tracking critical points in 2D time-dependent scalar fields, which is later used for the computations. Section 5.2 describes the different steps of the algorithm for the computation of the scale space persistence. Section 5.3 explores the properties of the new importance measure on a synthetic and a real-world dataset.

The contribution of this chapter is the novel importance measure for critical points, which is able to deal with data containing outliers in an effective way. The computation of it requires no preprocessing, the importance values are assigned to the critical points of the original data. The running time of the computation and the memory requirements allow to use scale space persistence in interactive systems.

The method presented in this chapter was published in [RKG+11]. This method was a result of a collaboration with Dr. Jan Reininghaus and Dr. David Günther, who were with Zuse Institut Berlin back then. For the implementation of the algorithm for the computation of Scale space persistence the implementations of other known algorithms were used (more details in Chapter 5). These implementations were a part of Amira software package, developed at Zuse Institut Berlin. The author of this thesis developed the original ideas, made research on the state of the art, especially in the area of Scale Spaces, which originates in Computer Vision and therefore not commonly known in the visualization community, participated in the implementation, analysis of the results and the preparation of the final publication.

Chapter 6 addresses a problem of a general visual representation of a topological structure, a 3D Morse-Smale complex. Section 6.1 describes a query language which helps to extract specific information out of the topological hierarchy. Section 6.2 discusses the issues on the geometrical representation of the components of the 3D Morse-Smale complex and presents an interactive topology rendering system in section 6.3. The example visualizations are presented and discussed in section 6.4.

The scientific contribution of this chapter is the presented topology visualization system, witch allows for the visualization of a 3D Morse-Smale complex in a flexible way. The topological structure is efficiently stored, the geometry for the visualization of the different components is extracted and a corresponding query language and the scene graph allow the user to interactively guide the visualization.

The visual system described in this chapter was published in [GKK+12]. This work resulted from a collaboration with the University of Utah, Scientific Computing Institute (SCI), an official partner of IRTG 1131, in particular with Dr. Attila Gyulassy. The algorithms for the computation of the topological structures used by this system were originally implemented by Dr. Gyulassy. For the rendering engine the system uses the modified implementation of the volume renderer, which was available as a part of the software package, developed at the SCI Utah, called ImageVis 3D. The author participated in the development of the ideas, the implementation of the system, in particular the graphical representation of the query language and the volume rendering of the geometry of topological structures, and the preparation of the final publication.

Chapter 7 presents two new methods for quality assessment of a triangulated surface. Section 7.1 motivates for the usage of surface interrogation methods. In section 7.2 the new technique to estimate the curvature values of a triangulated surface is described. Section 7.3 proposes a technique for computing the reflection lines on a tringulated surface, used for visual evaluation of the quality of the surface.

The contribution of this chapter are two new methods for:

- curvature estimation using local fitting with Bézier patches;
- computation of reflection lines.

These algorithms are especially beneficial for the visual assessment of the surface quality. This statement is supported by the comparisons and visual results presented in the chapter.

The contents of this chapter are published in [IGH+09]. The idea of this method originated before the author of this dissertation started her PhD at the University of Kaiserslautern. The original idea was immature and needed additional research work to be presented in a publication. The author reimplemented the algorithm for the curvature computation, made research on the state of the art and adapted the algorithm to be competitive with the known methods. The final presentation of the publication is the earliest work of the author of this thesis.

Chapter 8 concludes this thesis, presenting a summary of the research done and possible directions of future work.

Chapter 2

Theoretical background

This chapter provides the basic definitions of the concepts used later throughout this thesis. These concepts come from two main areas: differential topology and volume rendering. Only general well-known terms, which help the reader understand the later chapters better, are provided here. For more information the reader is referred to the corresponding literature, which will be described in detail in Chapter 3, unless not mentioned directly.

2.1 Differential topology

Topology is the branch of mathematics which studies shapes and spaces. A particular interest of topology lies in the properties invariant to continuous deformation. Well-known topological invariants are connectedness, continuity and boundary. Geometric topology is the study of manifolds and algebraic topology uses concepts from abstract algebra on topological spaces. This dissertation concentrates on both of these topics, giving special attention to the combinatorial solutions. Differential topology restricts the study to smooth functions on smooth manifolds. One of its branches, Morse theory, describes the topology of a manifold by the means of a smooth scalar function (Morse function) defined on it.

2.1.1 Basic definitions

To explain the terminology which is used throughout this work some definitions from basic and algebraic topology need to be stated first. Although the area of scientific visualization usually deals with the data of finite dimension, for the purpose of mathematical correctness the general definitions are described in this section. The material presented in this section is taken from various textbooks, the reader is referred to [Hat01, Arm10, Law06] for more information.

Definition 2.1.1. A *homeomorphism* is a function

$$H : X \to Y \tag{2.1}$$

which is a bijection and both H and H^{-1} are continuous mappings.

Definition 2.1.2. *Continuity.*

Let $B(z,r) ::= \{u \in \mathbb{R}^k : d(z,u) < r\}$ be a ball of radius r about $z \in R^k$.

$B_C(z,r) := C \cap B(z,r) = \{w \in C : d(w,z) < r\}$, where C is a subset of R^k. $C \cap B(z,r)$ consists of those points of C which are within distance r of z.
$f = X \to Y$ is *continuous at* $x \in X$ if given $\varepsilon > 0$, there is a $\delta > 0$ so that $B_X(x,\delta) \subset f^{-1}(B_Y(f(x),\varepsilon))$.

f is *continuous* if it is continuous at x for all $x \in X$.

Definition 2.1.3. *Open sets.*

- A set $U \subset \mathbb{R}^k$ is open if given any $y \in U$, then there is a number $r > 0$ so that $B(y,r) \subset U$.
- If X is a subset of $\mathbb{R}^k$ and $U \subset X$ then we say that U is open in X if given $y \in U$, then there is a number $r > 0$ so that $B_X(y,r) \in U$. Loosely speaking, U is an open set in X if it contains all of the points in X that are close enough to anyone of its points.

Remark 1. A set being open in a subspace does not need to be open in the whole space.

Definition 2.1.4. *Closed set.* A set $C \subset X$ is said to be *closed* if its complement $X \setminus C$ is open.

Example 1 *(Open set.).* In Figure 2.1.1 the grey area of the circle without the black boundary form an open set. It is a closed set, if the boundary is included.

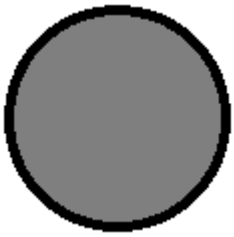


Figure 2.1.1: An open set and a closed set.

Theorem 2.1.1. *$f : X \to Y$ is continuous if the inverse image of an open set in Y is an open set in X.*

Definition 2.1.5. *Topological space.*

Let X be a set and let $\tau = \{M_i, i \in I\}$ be a collection of subsets of X. τ is called a *topology* on X. The sets M_i are called the open sets in the topology, if they satisfy:

1. the empty set and X are open sets
2. the union of any collection of open sets is open
3. the intersection of any finite number of open sets is open

$\{X, \tau\}$ is called a *topological space* if τ is a topology on X.

Two topological spaces $\{X_1, \tau_1\}$ and $\{X_2, \tau_2\}$ are said to be *topologically equivalent*, if there is a homeomorphism $H : X_1 \to X_2$.

Definition 2.1.6. *Subspace topology.*

Let $\{X, \tau\}$ be a topological space and $S \subset X$ be a subset of X. Then $\tau_s = \{S \cup U | U \in \tau\}$ is called *subspace topology* on S.

Definition 2.1.7. *n-manifolds.*

A topological space M is an n-manifold, if:

1. There is an embedding (a homeomorphism into its image with the subspace topology) of M into $\mathbb{R}^n$ for some n.

2. For each $x \in M$ there is a neighborhood U of x and a homeomorphism h from U onto an open set in $\mathbb{R}^n$.

Definition 2.1.8. *k-Simplex.*

Let $a^0, ..., a^k$ be $(k+1)$-affine independent points, then $S = conv\{a^0, ..., a^k\}$, where *conv* is a convex hull of the corresponding points, is called a k-dimensional simplex.

Every k-simplex is a k-dimensional manifold with boundary. It is topologically equivalent to a k-ball. Two simplices are homeomorphic to each other. This homeomorphism corresponds to the bijective ordering of their corners.

A *face* of a k-simplex is the convex hull of any nonempty subset of its $(k+1)$ points.

Definition 2.1.9. *Simplicial complex.*

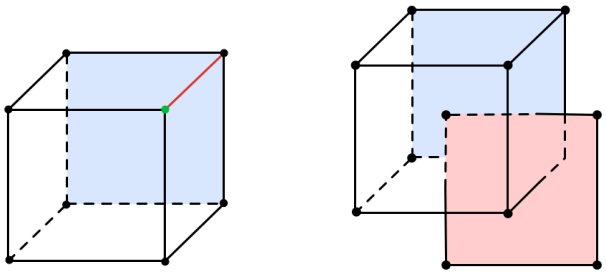
A *simplicial complex* K is a set of simplices satisfying the following conditions:

- Any face of a simplex in K is also in K;
- The intersection of any two simplices in K is either $\emptyset$ or their common face.

Let X be a topological space, K be a simplicial complex and $h : K \to X$ be a homeomorphism. Then K is called a *triangulation* of X.

This topological formulation of a triangulation is broader than a commonly known concept, where the term triangulation stands for two-dimensional triangles "glued" together. A three-dimensional manifold divided into tetrahedra is also called triangulation in the topological sense.

Simplicial complexes give a bridge from algebraic topology to the combinatorial one, allowing the computation on discrete grids. In the book of Edelsbrunner [EH10] there is a good introduction into the history and theory concerning this topic. This thesis uses a topological abstraction of a broader set of discrete meshes, namely CW complexes. A compact definition follows here and a combinatorial description of the methods will be stated later in this chapter and in the other chapters of this work.



(a) A 2-cell (blue) as a facet of a 3-cell; a 1-cell (red) and a 0-cell (green) as its faces.

(b) Improper operation of attaching a cell. This is not a cell complex.

Figure 2.1.2: Attaching a cell to a cell complex.

Let $B^d = \{x \in \mathbb{E}^d : |x| \leq 1\}$ be a closed unit ball in a d-dimensional Euclidean space. A *d-cell* σ is a topological space which is homeomorphic to B^d. If σ is a d-cell then $\dot{\sigma} \subset \sigma$ corresponds to $S^{(d-1)} = \{x \in \mathbb{E}^d : |x| = 1\} \subset B^d$ (the boundary of B^d) under any homeomorphism between B^d and σ.

Let α and β be two cells. α is a *face* of β or $\alpha < \beta$ if α is a proper subset of β. In this case β is a *co-face* of α. If the dimensions of α and β differ only by 1, they are called *facet* and *co-facet* of each other respectively.

Let X be any topological space and $f : S^{(n-1)} \to X$. The operation of *attaching a cell* σ *to* X, denoted by $\sigma \cup_f X$ is done by identifying each point of $S^{(n-1)}$ with its map $f(x)$ on X (attaching a cell along its boundary, see Figure 2.1.2).

A *cell space* is a topological space obtained from a finite set of points by iterating the procedure of attaching cells of arbitrary dimensions, with the condition that only finitely many cells of each dimension are attached.

A cell space X is called a *cell complex* (CW-complex) if each cell is attached to cells of lower dimension. The *n-skeleton* X_n of X is constructed from all cells of dimension $k \leq n$. A cell complex X can then be represented as

$$X_0 \subset X_1 \subset \cdots \subset X_n \subset \cdots \subset X.$$

2.1.2 Morse theory

Morse theory characterizes the topology of an n-dimensional Rimmanian manifold (n-manifold) through the behavior of a differentiable function f in its critical points, assuming that f has only non-degenerate critical points.

The following definitions are due to [Mil63, Mat01].

Definition 2.1.10. Let f be a smooth real-valued function on a manifold M. A point $p \in M$ is called a *critical point* of f if the differential of f vanishes at this point.

A critical point is called *non-degenerate* if the Hessian matrix (the matrix of second partial derivatives) of f is non-singular at this point. Otherwise a critical point is called *degenerate*.

Example 2. Monkey saddle.

$f(x,y) = x^3 - 3xy^2 =$ Real part of $(x+iy)^3$. $p = (0,0)$ is a degenerate critical point (Fig. 2.1.3).

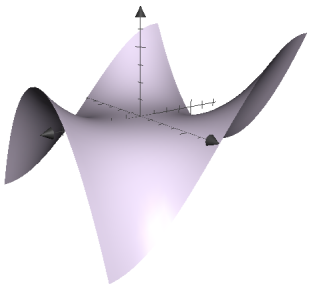


Figure 2.1.3: A Monkey saddle

Definition 2.1.11. The *index* of a non-degenerate critical point p of f is the dimension of the largest subspace of the tangent space to M at p on which the Hessian matrix is negative definite.

More intuitively, the index of a critical point is the number of negative values of the Hessian matrix at this point, i.e. the number of independent directions around this point in which f decreases. Let us take a two-dimensional manifold (a surface) as an illustrative example. If p is a maximum, it means the function decreases all around p, but there are

only two independent vectors on a tangent plane to this point, therefore the index of criticality of a maximum in this case is 2 (in general, a maximum in any dimensions gets the maximum possible index, which is equal to the dimensionality of a manifold). Now imagine a minimum, around which f can only increase. It's index is always 0. At a non-degenerate saddle point f decreases in one direction and increases in another. It's index is 1. For manifolds of dimensions higher than two one distinguishes between different kinds of saddles, according to their indices.

Definition 2.1.12. A smooth map $f : M \to \mathbb{R}$ is a *Morse function* if none of its critical points are degenerate and no two critical points have the same function value.

Example 3. Morse function.

Consider the unit sphere S^2 in $\mathbb{R}^3$ with orthogonal coordinates (x,y,z):

$$x^2 + y^2 + z^2 = 1.$$

Then $f : S^2 \to \mathbb{R}$, $f(x,y,z) = z$ (the height function) is a Morse function. f has two critical points $p_0 = (0,0,1)$ and $p_1 = (0,0,-1)$, both of them are non-degenerate.

Lemma 2.1.1. Morse lemma.

Let p be a non-degenerate critical point of a Morse function f. Then there is a local coordinate system $(y^1,...,y^n)$ in a neighbourhood U of p with $y^i(p) = 0$ for all i and such that the identity $f = f(p) - (y^1)^2 - (y^\lambda)^2 + (y^{\lambda+1})^2 + (y^n)^2$ holds throughout U, where λ is the index of f at p.

Because scientific visualization usually represents objects and phenomena from the real three-dimensional world, this thesis mainly deals with functions defined on 2- and 3-manifolds. The following lemma is a special case of the Morse lemma for 2-dimensional manifolds, which are called surfaces. The naming of critical points depending on their index will be used in the later chapters of this work.

Lemma 2.1.2. Morse lemma for surfaces.

Let p be a critical point of a Morse function f defined on a surface. Then one can choose appropriate local coordinates (x,y) with p as the origin in such a way that the function f expressed with respect to (x,y) has one of the following 3 standard forms:

1. $f = x^2 + y^2 + c$ *(index of p is* 0*, minimum),*
2. $f = x^2 - y^2 + c$ *(index of p is* 1*, saddle),*
3. $f = -x^2 - y^2 + c$ *(index of p is* 2*, maximum),*

where $c = f(p)$.

Definition 2.1.13. An *integral line* of f is a maximal path in M whose tangent vectors agree with the gradient of f at every point.

Integral lines connect critical points. The set of integral lines sharing the common origin/destination is called an *ascending/descending manifold*[1]. A function is called *Morse-Smale* if its ascending and descending manifolds intersect only transversally.

2.1.2.1 Morse-Smale Complex

The *Morse-Smale complex* is a fundamental topological construct that partitions the domain of a real-valued function into regions having uniform flow behavior, namely the regions, where integral lines share common origin and destination. A Morse-Smale complex is a graph whose *nodes* are represented by critical points of f, *arcs* are obtained by superimposing the ascending and descending 1-manifolds of all critical points and *regions* are 2-dimensional cells obtained by intersection of descending and ascending 2-manifolds. An example of a simple Morse function is shown in Figure 2.1.4, where one can see the visual representation of the Morse-Smale complex and its underlying structures. A more complete visual guide to Morse theory is presented in [GKK$^+$12].

Compared to other known topological structures, like contour trees or Reeb graphs, a Morse-Smale complex describes the topology of the underlying manifold most fully, giving an additional information about the gradient flow. That makes it useful for purposes like segmentation of the transfer function domain, described in Chapter 4. Later in Chapter 6 the Morse-Smale complex is visualized allowing for the interactive topology exploration.

[1] Sometimes in literature, for example in [EH10], ascending/descending manifolds are called stable/unstable manifolds. This work uses the former notation.

For more information about continuous Morse theory the reader is referred to [Mil63, Mat01].

2.1.3 Discrete Morse Theory

Discrete Morse theory is a combinatorial equivalent of the continuous Morse theory. It was founded by Forman [For01], therefore it is sometimes called Forman's discrete Morse theory. This section presents a few theoretical concepts needed to establish the notation for further chapters of this thesis.

Let K be a cell complex as defined in Section 2.1.1. A *Discrete Morse function* $f : K \to \mathbb{R}$ is a function which assigns scalar values to all cells in K, such that for every cell α:

1. the number of co-faces of α with function value lower than $f(\alpha)$ is at most one;
2. the number of faces of α with function value higher than $f(\alpha)$ is at most one.

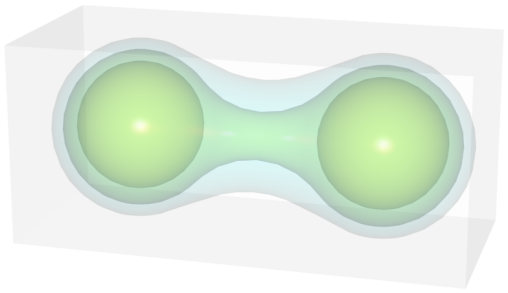
A cell is *critical* if all of its co-faces have a function value higher than $f(\alpha)$ and all of its faces have a function value lower than $f(\alpha)$.

A *discrete vector field* is a collection of pairs of cells $\{\alpha^{(d)} < \beta^{(d+1)}\}$ ($\alpha^{(d)}$ is a facet of $\beta^{(d+1)}$, see Section 2.1.1) such that each cell is in at most one pair. Each pair represents a *discrete vector*, where the vector arrow points from $\alpha^{(d)}$ to $\beta^{(d+1)}$. A *gradient vector* in a discrete Morse function f is a pair of cells $\{\alpha^{(d)} < \beta^{(d+1)}\}$ for which $f(\alpha^{(d)}) \geq f(\beta^{(d+1)})$. According to the definition of the discrete Morse function each cell can have at most one facet with higher function value, this is why the discrete gradient is uniquelly defined for each cell. Critical cells according to their definition can not be paired. This simulates the vanishing gradient for these cells.

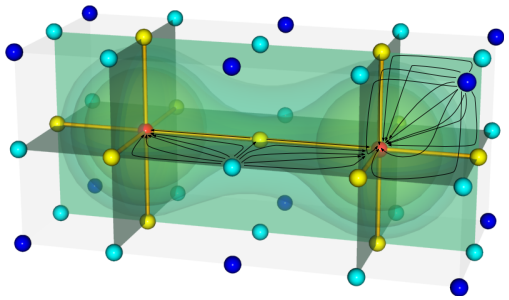
A *V-path* is a sequence of cells:

$$\alpha_0^{(d)}, \beta_0^{(d+1)}, \alpha_1^{(d)}, \beta_1^{(d+1)}, \alpha_2^{(d)}, ..., \beta_r^{(d+1)}, \alpha_{r+1}^{(d)},$$

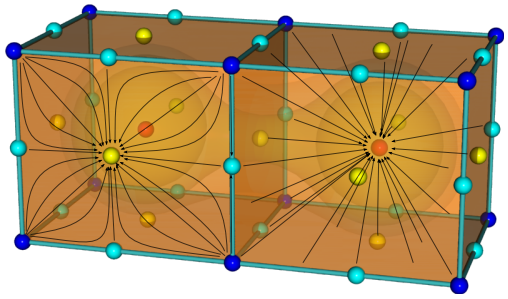
where $\{\alpha^{(d)} < \beta^{(d+1)}\} \in V$, and $\{\beta_i^{(d+1)} > \alpha_{i+1}^{(d)} \neq \alpha_i^{(d)}\}$. This path is closed if $\alpha_0 = \alpha_{r+1}$. V-paths in a discrete gradient field are monotonic and contain no closed V-paths, because by definition of discrete gradient vector (see above) $\alpha_0^{(d)}$ is the facet of $\beta_0^{(d+1)}$



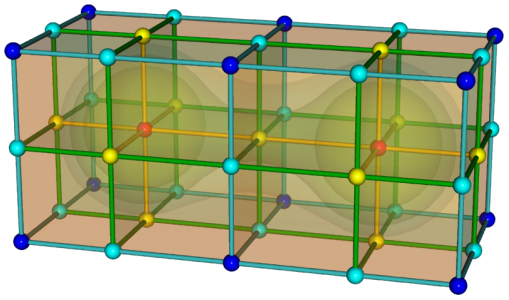
(a) **Morse function:** This function is a sum of two Gaussians. The function value is highest in two points bounded by spherical contours.



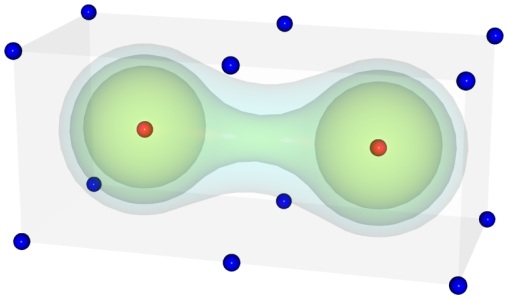
(b) **Ascending Manifold:** Group of integral lines sharing a common origin. Gradient lines starting from an index-i critical point form an i-manifold.



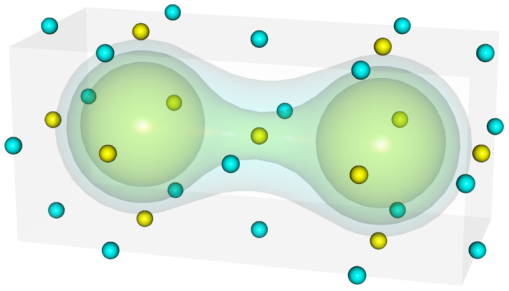
(c) **Descending Manifold:** Group of integral lines sharing a common destination. In dimension d gradient lines ending in an index-i critical point form a $(d-i)$-manifold.



(d) **Morse-Smale complex:** Partition of the space in cells computed as intersection of the ascending and descending manifolds. It forms the basis of a large feature space.



(e) **Maxima/Minima:** At a maximum (red) the function has locally highest value. At minimum (blue) the function has locally lowest value.



(f) **Saddles:** At a saddle gradient flow diverges. A contour swept through a saddle value changes its topology.

Figure 2.1.4: Topological definitions in the scope of Morse theory [GKK+12].

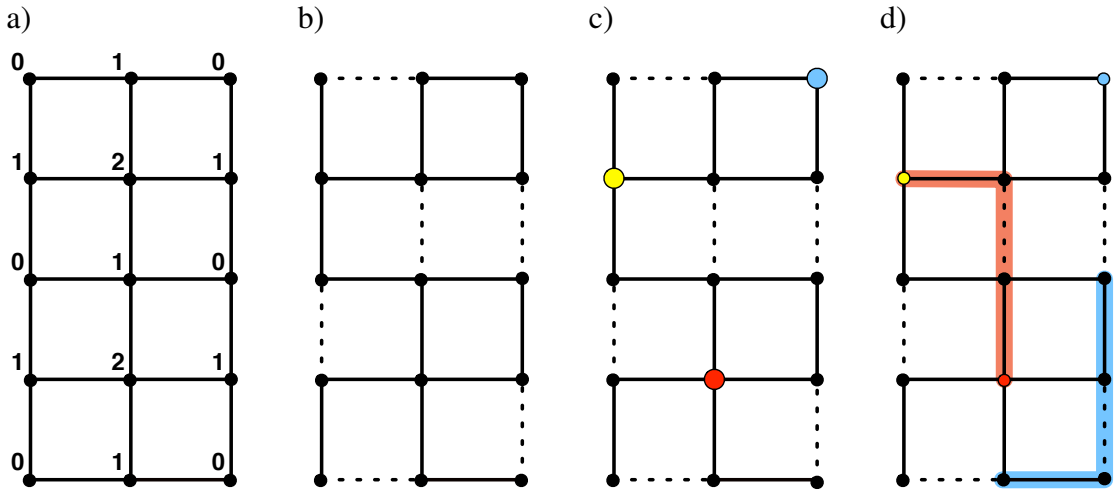


Figure 2.1.5: Basic definitions of discrete Morse theory in a graph-theoretic setting illustrated on a 2×1 grid: a) the cell graph G, the node labels indicate the dimension of the represented cells; b) a combinatorial gradient field V defined on G, the edges contained in V are depicted by dashed lines; c) the critical points of V, indicated by colored circles (minimum = blue, saddle = yellow, maximum = red); d) a 0-streamline of V (blue) and a 1-streamline of V (red).

with the highest function value, i.e. $f(\alpha_0^{(d)}) > f(\alpha_1^{(d)})$, where $\alpha_1^{(d)}$ is another facet of $\beta_0^{(d+1)}$, f decreases monotonically from α_0 to α_{r+1}.

To define a discrete Morse-Smale complex the combinatorial equivalents of ascending and descending manifolds have to be explained. The following definitions are due to [Gyu08].

The *boundary operator* maps a cell to its facets and induces an orientation on the facets. The *discrete tangent operator* $T(\beta)$ maps the cell at the tail of a gradient arrow to the head of the gradient arrow. The *discrete flow operator* Φ is defined as $\Phi(\alpha) = \alpha + \partial T(\alpha) + T(\partial\alpha)$. The *discrete descending manifold* D_α of a critical cell α is an invariant chain under the flow operator $\Phi(D_\alpha) = D_\alpha$. The *discrete ascending manifold* can be defined analogously. The *discrete Morse-Smale complex* of a cell complex K is formed by the intersection of its descending and ascending manifolds, as defined in a continuous setting, see Section 2.1.2.1.

Based on the Forman's theory Rheininghaus et al. [RGH+10] defined combinatorial gradient field in a graph theoretic setting. The essential combinatorial information of a cell complex is encoded in the graph $G = (N, E)$, where the nodes N of the graph consist of

the cells of the complex and each node u^p is labeled with the dimension p of the cell it represents. The edges E of the graph encode the neighborhood relation of the cells. If the cell u^p is in the boundary of the cell w^{p+1}, then $e^p = \{u^p, w^{p+1}\} \in E$ (see Figure 2.1.5a). Additionally each edge is labeled with the dimension of its lower dimensional node.

A subset of pairwise non-adjacent edges is called a matching. A *combinatorial gradient field* $V \subset E$ on a regular cell complex C can now be defined as a matching of the cell graph G (see Figure 2.1.5b) with a certain acyclic constraint, see below. The unmatched nodes of V are called critical points. If u^p is a critical point, the critical point is said to have index p. A critical point of index p is called minimum $(p = 0)$, saddle $(p = 1)$, or maximum $(p = 2)$ (see Figure 2.1.5c). A combinatorial p-streamline is a path in the cell graph whose edges are of dimension p and alternate between V and its complement $E \setminus V$ (see Figure 2.1.5d). The above mentioned acyclic constraint can now be specified as the non-existence of any closed p-streamline.

2.1.4 Topology simplification and homological persistence

A commonly used importance measure for critical points in the discrete setting is homological persistence. This measure is based on a certain pairing of critical points. For simplicity the definition of this pairing is restricted to the 1D case and an interested reader is referred to [ELZ02] for the general case.

For a 1D function f this measure can be defined by considering the number of components of the sublevel sets $F_r = f^{-1}((-\infty, r])$. As r increases the number of components in F_r changes: when r passes the value of a local minimum a component is born, while two components merge when r passes the value of a local maximum. A maximum is then paired with the larger minimum of the two merged components. Persistence of the paired critical points is now defined as the difference of their function values. Note that paired critical points are not necessarily adjacent. Figure 2.1.6 shows a simple example illustrating the concept of homological persistence.

Persistence-based topology simplification is a common technique to reduce the feature space and leave only the ”most important” (in the sense of homological persistence) features in the domain. In the Morse theory application area the features are critical points, each of which gets its persistence value assigned. The simplification process consists of

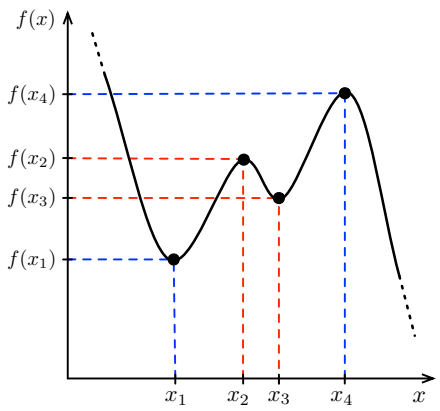


Figure 2.1.6: Homological persistence of a 1D function $f(x)$. The persistence pairs consist of (x_1, x_4) and (x_2, x_3). The persistence of x_1 and x_4 is therefore given by $f(x_4) - f(x_1)$, while the persistence of x_2 and x_3 is given by $f(x_3) - f(x_2)$.

a pairwise canceling of critical points, those with the lowest value first. Besides noise reduction this kind of simplification aims for better perception of the data removing the ”unimportant” features.

2.2 Volume rendering

Volume rendering is a range of techniques for visualizing spatial scientific data. Although volume rendering can be used to generate images from any data, regularly sampled on a 3D grid, most often it is aimed for the visualization of medical data, coming from Computed Tomography or Magnetic Resonance Imaging. Seismic data, Computational Fluid Dynamics simulations or other scalar fields are also typical applications for volume rendering techniques. Due to the development on the algorithmic side and recent advances in the graphics hardware volume rendering application area went beyond scientific visualization. Special effects and photorealistic rendering got much attention in the game and movie industry, opening the doors for volume rendering into the area of visual arts.

This section presents some basic theory behind the process of volume rendering and especially its most widespread technique, ray casting. This presentation is based on the information from [HKRs+06], which is also recommended as an additional source for more details on the topic.

2.2.1 Direct volume rendering

The purpose of direct volume rendering is to visually extract information from a scalar field. A scalar field is represented by a mapping

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}^N, \tag{2.2}$$

which maps each point of a 3D dataset to some scalar value(s). If $N = 1$, one speaks of univariate data, if $N > 1$, the dataset is called multivariate.

The scalar values of the original dataset represent some physical identity depending on the nature of the data. The process of assigning the optical properties to each point in the dataset depending on the scalar value is called *classification*. The mapping used for classification is called *transfer function*:

$$\rho : \mathbb{R}^N \rightarrow \text{RGBA},$$

where RGBA represents the optical properties corresponding to each scalar value.

The choice of the appropriate transfer function usually depends on the application domain and is in general not a trivial task.

2.2.2 Volume rendering integral

The volume rendering pipeline is based on the physical properties of light propagation. To identify the color and opacity at some point in the volume, the term of the light *radiance* at this point is used. The radiance is defined as:

$$I = \frac{dQ}{dA_\perp d\Omega dt}, \tag{2.3}$$

where Q is the radiative energy per unit area $A_\perp$ (projected along the light direction), angle Ω and time t.

To construct the light transport equation one has to take into account different manners in which a material can interact with light: emission, absorption, scattering.

The physically complete version of the light transport equation is

$$\begin{aligned}\omega \cdot \nabla_{\mathbf{x}}(\mathbf{x},\omega) = &-(\kappa(\mathbf{x},\omega)+\sigma(\mathbf{x},\omega))I(\mathbf{x},\omega)+ \\ &q(\mathbf{x},\omega)+\int_{sphere}\sigma(\mathbf{x},\omega')p(\mathbf{x},\omega',\omega)I(\mathbf{x},\omega')d\Omega'\end{aligned} \tag{2.4}$$

where x is the point position in the volume, ω is the ray direction, ω' are all possible directions, κ is the absorption coefficient, σ is the scattering coefficient, q is the emission coefficient and I is the radiance. The coefficients κ, σ and q are usually specified by the transfer function. The phase function p is the probability that the light is scattered into the new direction.

Photorealistic rendering is rarely used for scientific visualization applications, which usually aim at real time interaction. The equation 2.4 is performance hampering if evaluated for every point in the volume. Therefore some tradeoffs have to be accepted in order to increase the calculation speed. The optical model which got popular in the area of scientific visualization is called *emission-absorption model*. In this model the scattering and indirect illumination are neglected, which allows the reduction of equation 2.4 to:

$$\omega \cdot \nabla_{\mathbf{x}}(\mathbf{x},\omega) = -\kappa(\mathbf{x},\omega)I(\mathbf{x},\omega)+q(\mathbf{x},\omega). \tag{2.5}$$

Equation 2.5 is called the *volume rendering equation*. The differential form of the previous equation for one ray can be written as:

$$\frac{dI(s)}{ds} = -\kappa(s)I(s)+q(s), \tag{2.6}$$

where s is the length parameter.

The solution of the above differential equation is called the *volume rendering integral*:

$$I(D) = I_0 e^{-\int_{s_0}^{D}\kappa(t)dt}+\int_{s}^{D} q(s)e^{-\int_{s}^{D}\kappa(t)dt}ds \tag{2.7}$$

The main purpose of any volume rendering algorithm is to approximate the volume rendering integral.

2.2.3 Classification and transfer functions

The simplest one-dimensional transfer function can be represented by a color table, which assigns a color value to every data value. However the application of this kind of transfer functions is very limited. By specifying a transfer function the domain expert aims to highlight the important features and hide the unimportant ones, based on the range of some abstract data values.

Transfer function specification is a domain-dependent task. The usual way to give the domain expert the flexibility of choosing a transfer function is to provide a transfer function widget, which allows for an interactive adjustment of the function. Because of this interactivity requirement one of the challenges of direct volume rendering is its performance. There are approaches for the automatic transfer function definition ([KD98, KWTM03, TLM03, HM03]), but at the moment they can not give the desired flexibility.

As mentioned earlier, interpolation is needed to reconstruct a continuous 3D scalar field from a given discrete data. There are two ways to apply an interpolation scheme in the volume rendering pipeline: interpolate directly between data values and apply a transfer function to the interpolated data (*post-classification*) or interpolate between the color values after the transfer function application (*pre-classification*). These two classification arts usually give different results, depending on the data and the transfer function. It is proven (see [HKRs+06]) that pre-classification gives more rendering artifacts than post-classification. The reason for this is that the transfer function application introduces additional high frequencies to the data before interpolation. According to the sampling theorem, the data has to be sampled finer, for pre-classification to give the same visual results as post-classification. There are exceptions to this rule, for example, when the discrete edges of the segments of the data are desired. In general, for the volume rendering of scientific data, post-classification is usually used.

Depending on the data, the important and unimportant (object/background) value ranges might intersect, making the classification with a color table an impossible task. This is often the case for the medical data. This challenge moved the science into the direction of *multidimensional transfer functions*. In contrast to a one-dimensional transfer function, which maps one scalar value to the optical properties: color and opacity, a multidimensional transfer function takes more information as input, usually several scalar values.

All different kinds of derived values can be used as an additional dimension in a transfer function. The *gradient-based* transfer functions, which take the gradient magnitude as the second dimension, got the largest application in scientific visualization. Because of the discrete nature of the data and the need of the interpolation between given values one can not get sharp boundaries between objects. The ”smoothed” boundary values may fall into the value range of a different object, highlighting an undesired area. If one takes into account not only the scalar field value itself, but its gradient magnitude too, it allows to avoid the described problem. The areas of a highest gradient magnitude are those of the boundary of an object.

Multidimensional transfer functions offer themselves naturally for multivariate data, where each data sample is assigned multiple scalar values. Each dimension in a transfer function corresponds than to a ”channel” in the dataset, allowing to highlight the features, taking into account all the data properties. An example for this is a simulated 2-variate computational fluid dynamics dataset, which will be used later in this thesis (see Figure 4.2.4 in Section 4.2.3), with vorticity and normalized helicity values assigned to each point in the volume. Analogously to 1D transfer functions, a multidimensional transfer function is usually a lookup table, uniquely specifying optical properties for each combination of the dimensional values.

Multidimensional transfer functions not only increase the memory usage, they are also much harder to design. The transfer function widgets nowadays use all kinds of user interactions: from ”pulling” on a height field in 3D to ”drawing” the simple geometric shapes in 2D. The domain expert who tries to specify such a function has to spend more time doing this than in the case of 1D transfer functions. The research on multidimensional transfer functions gets more and more attention lately. The reader will be introduced to the contribution of this thesis in this area in Chapter 4.

2.2.4 Preintegration

Preintegration [EKE01] is one of the most common methods nowadays to improve on the approximation of the volume rendering integral. Its main idea is to precompute color and opacity values for each linear segment along the ray and then linearly interpolate between the precomputed values. It helps to avoid the undersampling resulting from the

combination of the Nyquist frequencies of the scalar field and the transfer function. The approximation of the volume rendering integral used for preintegration is:

$$I \approx \sum_{i=0}^{n} c_i \prod_{j=0}^{i-1} (1-\alpha_i), \tag{2.8}$$

where c_i are the color values for the i-th segment:

$$c_i \approx \int_0^1 q((1-\omega)s_f + \omega s_b) \times e^{-\int_0^\omega \kappa((1-\omega')s_f+\omega' s_b)d\mathrm{d}\omega'} \mathrm{d}\omega \tag{2.9}$$

and α_i are the opacity values:

$$\alpha_i \approx 1 - e^{-\int_0^1 \kappa((1-\omega)s_f+\omega s_b)d\mathrm{d}\omega} \tag{2.10}$$

One of the disadvantages of preintegration is a relatively long precomputation time, which sometimes does not allow the transfer function modification at interactive rate. The memory consumption due to the precomputed structures is also a trade-off for the better rendering quality. Lookup tables can become of a very large size, which increases the minimal requierement on graphical RAM to perform this method.

To accelerate the computation of the approximation of the volume rendering integral Engel et.al. [EKE01] proposed to make use of the integral functions $G(s)$:

$$G(s) = \int_0^s f(x)\mathrm{d}x \tag{2.11}$$

An integral $F(x)$ can then be written as:

$$F(x) = \int_a^b f(x)\mathrm{d}x = G(b) - G(a) \tag{2.12}$$

Equation 2.10 can be rewritten as:

$$\alpha_i \approx 1 - e^{-\frac{d}{s_b - s_f}(T(s_b)-T(s_f))}, \tag{2.13}$$

where

$$T(s) = \int_0^s \kappa(s')\mathrm{d}s' \tag{2.14}$$

Equation 2.9 can be rewritten correspondingly as:

$$c_i \approx \frac{d}{s_b - s_f}(K(s_b) - K(s_f)), \tag{2.15}$$

where

$$K(s) = \int_0^s q(s')\mathrm{d}s'. \tag{2.16}$$

To speedup the rendering process Engel et al. propose to precompute the values c_i and α_i for all possible scalar values and save them in a 2D lookup table. Obviously, the lookup table can become very large, depending on the original data. Although, an extension of this technique to multidimensional transfer functions has been proposed [Kra08], its memory requierements for using higher dimensional structures and computation time make it impractical to use.

Chapter 3

State of the art

This chapter contains a review of the research done in the areas of topology-based visualization and volume rendering. These are the most relevant fields for the topic of this thesis. Later chapters briefly refer to the state of the art in other computer science areas, the relevance of which is local for one chapter only.

3.1 Topology-based visualization

Topology-based techniques are well-known in the context of scalar function analysis. In general, a topological representation of the function is computed, then queried. Reeb graphs [Ree46], contour trees, and their variants have been used successfully to guide the removal of topological features [CCM+00, CSvdP04, GW01, TNTF04, WHDS04, CSA03, CLLR05]. Pascucci et al. [PSBM07] showed how the Reeb graph can be constructed in a streaming manner for large datasets. Tierny et al. [TGSP09] presented an efficient algorithm to compute Reeb graphs by cutting the domain, and showed how it could be used for isosurface simplification. Weber et al. [WDC+07] used contour trees to segment the domain and render each region with a separate transfer function. Chiang and Lu used the augmented contour tree to simplify tetrahedral meshes while preserving isosurface topology [CL03].

Partitions of surfaces induced by a piecewise-linear (PL) function have been studied in different fields, under different names, motivated by the need for an efficient data structure

to store surface features. Cayley [Cay59] and Maxwell [Max70] proposed a subdivision of surfaces using peaks, pits, and saddles along with curves between them. The development of various data structures for representing topographical features was discussed by Rana [Ran04].

As known from Chapter 2, *The Morse-Smale complex* is a topological data structure that provides an abstract representation of the gradient flow behavior of a scalar field [Sma61b, Sma61a]. Several algorithms have been proposed to compute MS complexes in practice: Edelsbrunner et al. [EHZ03] presented the first algorithm for two-dimensional data, and Bremer et al. [BEHP04] improved this by following gradients more faithfully and described a multi-resolution representation of the scalar field. Although an algorithm was proposed to compute all dimensional manifolds of a three-dimensional complex [EHNP03], a practical implementation was never done due to the complexity of the algorithm. The one-skeleton (0- and 1-manifolds) of the MS complex was first computed successfully for volumetric data by Gyulassy et al. [GNP$^+$05]. Although the same authors presented a more efficient approach to computing the MS complex by using a sweeping plane [GNP$^+$06], data size and computational overhead still proved to be a limiting factor.

Discrete Morse theory, as presented by Forman [For01], is an attractive alternative to PL Morse theory, since it simplifies the search for higher dimensional manifolds by discretizing the flow operator. The challenge in using a discrete approach is in generating a discrete gradient field. Lewiner et al. [LLT04] showed how a discrete gradient field can be constructed and used to identify the MS complex. However, this construction required an explicit graph-based representation of gradient paths, thus, was prohibitively expensive for large volumetric data. King et al. [KKM05] presented a method for constructing a discrete gradient field that agrees with the large-scale flow behavior of the data defined at vertices of the input mesh. Gyulassy et al. [GBHP08] proposed a scalable method for the computation of the 3D Morse-Smale complexes. A method for tracking critical points in scalar fields based on the gradient vector fields was developed by Reininghaus et al. [RKWH11].

These techniques have begun to make an impact in analysis of scientific data. Laney et al. [LBM$^+$06] used the descending 2-manifolds of a two-dimensional MS complex to segment an interface surface and count bubbles in a simulated Rayleigh-Taylor instability. Bremer et al. [BWP$^+$10] used a similar technique to count the number of burning regions

in a lean premixed hydrogen flame simulation. Gyulassy et al. [GDN+07] used carefully selected arcs from the 1-skeleton of the three-dimensional MS complex to analyze the core structure of a porous solid. In each of these examples, feature definitions were attained by exploring thresholds, levels in a hierarchy, and different components of the MS complex, clearly illustrating the need for interactive feature exploration. Although each result was obtained from different implementations, they all use the same underlying mathematical theory, motivating a unified query system for topological structures. Such a system will be presented in Chapter 6 of this thesis.

As known from Chapter 2, *homological persistence* is an algebraic method for measuring the importance of critical points. The basic concepts have been independently developed in [FL99, Rob99, ELZ02]. A good survey of the state of the art in the area of persistent homology can be found in [EH10]. A strong stability result for the persistence measure has been proven in [CSEH07]. It has also been shown that persistence can be used to simplify the Morse-Smale complex [Zom01] of a 2D scalar-valued function. Later Bauer et al. [BLW10] has proven this resunt not to hold in higher dimensions. Homological persistence is a powerful measure for topology simplification if the data does not contain outlier-like noise. The persistence values of the outliers are very high, therefore they are not being removed on the early stages of simplification, which makes the method sensitive to this kind of noise. In Chapter 5 the new importance measure, Scale space persistence, is presented, which takes into account the spatial extent of a critical point, assigning lower values to ”thin peaks” (outliers). The outliers are removed on the early stage of the simplification.

3.2 Volume rendering

Modern volume rendering techniques have been used for the past thirty years. Kajiya and Von Herzen [KVH84] used ray tracing to render objects represented by densities within a volume grid, based on light scattering equations. Drebin et al. [DCH88] and Levoy [Lev89] laid out the foundation for volume rendering, such as compositing methods, gradient-based shading, and volume classification. Max [Max95] discussed several different models for light interaction with volume densities, including absorption, emission, reflection, and scattering. Fast rasterization hardware made interactive direct volume

rendering possible with slicing [CN94, CCF94]. While splatting [Wes90] is feasible, ray casting [KW03, RGW+03] has regained popularity due to its efficient implementation on current GPU hardware. Isosurface mesh extraction from structured volume data was first proposed by [LC87] and remains a common method for visualization. Direct ray casting of isosurfaces was proven on the CPU [Sra94, PSL+98] and later on the GPU [HSS+05]. Kraus [Kra05] reformulated direct volume rendering as an integration of isosurfaces, showing that irradiance can be computed without normalizing the Riemann sum over the number of samples. Multifield isosurface rendering has been used to visualize particle astrophysics data. Navratil et al. [NJB07] use marching cubes to extract separate meshes, while Linsen et al. [LLRR08] employ a particle reconstruction method to resample and render a single surface from multiple channels.

Volume classification provides the means for visually segmenting volumetric data. Kindlmann and Durkin [KD98] proposed the histogram volume, which captures the relationship between volumetric quantities in a position independent, computationally efficient fashion. They presented semi-automatic methods of generating transfer functions for direct volume rendering. Correa and Ma [CM08, CM09] proposed visibility-driven and size-based transfer function design techniques for volume exploration. While transfer function design with multi-dimensional histograms can discriminate between features (materials) in the data it requires a strong background knowledge about the data and is time consuming. Methods have been proposed to accelerate the transfer function design process. Rezk-Salama et al. [RSKK06] introduced an additional level of abstraction for parametric models of transfer functions, using semantic models for transfer function design. Tzeng et al. [TLM05] proposed an approach to the volume classification problem that couples machine learning and a painting metaphor to allow more sophisticated classification in an intuitive manner.

Section 2.2.3 discussed the advantages of using the *multidimensional transfer functions* for some applications. Laidlaw [Lai95] first advocated multidimensional transfer functions for improved classification of MRI data. 2D transfer functions with gradient magnitude [KKH02, Kin02] or curvature [KWTM03] can greatly improve classification flexibility, particularly for noisy scan data in biology and medicine. Kniss et al. [KPI+03] applied specially constructed Gaussian kernels to multifield volume visualization using an analytical integration method for improved visual quality. Simpler classification and

blending operations, such as maximum-intensity projection, can equally be used for rendering multifield data [SR04].

Section 2.2.4 has briefly reviewed *preintegration* for one-dimensional transfer functions. It was proposed by Roettger et al. [RKE00] and extended to trilinearly-interpolated structured volumes by Engel et al. [EKE01]. Preintegration integrates transfer function space using a separate Riemann sum. Irradiance on a ray segment can then be queried in a 2D lookup table. An extension of this technique to multidimensional transfer functions has been presented in [Kra08]. Summed area tables are used for performing the preintegration and rendering. To compensate for the artifacts produced by this method, they propose casting beams as opposed to discrete rays, gathering correspondingly wider integrals in transfer function space. The process of rendering becomes computationally expensive. It is also costly to preintegrate high-resolution 2D transfer functions, and this approach has not been extended to higher-dimensional classification. Peak finding [KHW+09] combines direct volume rendering with discrete isosurfacing by sampling directly at peaks in the transfer function. Ament et al. [AWC10] detail a more robust method for DVR integration of discrete isosurfaces that removes scale-dependency entirely. However, it is expensive (requiring 3 samples per voxel as opposed to multiple voxels between samples for peak finding) and would not extend easily to multidimensional classification due to its reliance on lookup tables.

The volume rendering method introduced in Chapter 4, Multidimensional peak finding, extends the one-dimensional peak finding idea to the multidimensional transfer functions. It overcomes the computational complexity of the multidimensional preintegration and allows for the higher-dimensional classification with non-separable transfer functions.

Chapter 4

Volume rendering with multidimensional peak finding

As was discussed in Section 2.2, the purpose of direct volume rendering is to produce images of a high visual quality, highlighting the features of interest, desirably at an interactive rate. Approximating a volume rendering integral is a computationally expensive task, which heavily depends on the sampling rate along a ray. According to [BMWM06] the optimal sampling rate for artifact-free rendering depends on the Nyquist frequency of the scalar data and of the transfer function, therefore transfer functions with high frequency features have always been a challenge for known volume rendering methods. Such transfer functions are useful, for example, for isosurfacing, where the transfer function contains one or several thin peaks. Although preintegration solves this problem partially, by sampling the volume data and the transfer function separately, it is limited by its initial assumptions and is unable to produce visually compelling results for this case. First proposed by Knoll et al. [KHW+09], peak finding addresses this problem by identifying the peak along the ray and sampling exactly at it, lowering the requirements for the initial sampling rate of the scalar data. It has proven useful for noisy data, where the features, introduced by noise, are undesirable.

4.1 One-dimensional peak finding

The main idea of the peak finding method is not to allow the peaks in the transfer function to be missed. Peaks in transfer functions designed to show isosurfaces usually fall in-between sampled values and therefore are incorrectly interpolated. Searching for a transfer function peak along each segment of the ray and sampling exactly at its location allows to avoid this problem. If v is a desired isovalue and $f(\vec{R}(t))$ is the scalar function value at a sampled point along the ray $\vec{R}$, then the equation which has to be solved for parameter t, can be written as:

$$f(\vec{R}(t)) - v = 0. \tag{4.1}$$

Similar to the preintegration method, a 2D lookup table is constructed in the preprocessing step. After the local maxima analysis of the transfer function, the domain isovalues v_i are stored for each ray segment, corresponding to the peaks in the transfer function space. The constructed lookup table is then used in the main ray casting loop. By comparing the values at the beginning and end of each ray segment with the corresponding table entry v_i, it is decided if the segment contains v_i. If so, the secant method is employed to find the exact position t of this value along the ray $\vec{R}$. An additional sample is then put exactly at this position.

In their original paper Knoll et al. have proven the method qualitatively better than the postclassification and preintegration. The performance of the method heavily depends on the scalar field and the transfer function. Multiple peaks spaced close together slow down the algorithm a little, but this situation appears rarely in practice.

4.2 Multidimensional peak finding

The extension of a one-dimensional technique to multiple dimensions is not as straightforward as it appears at first glance. Peaks along a world-space ray almost never occur at maxima in higher-dimensional transfer function space, as shown in Figure 4.2.1. In most cases, therefore, one cannot precompute a peak table and solve for an isovalue at each peak along a ray segment. When handling multidimensional transfer functions, different strategies must be used.

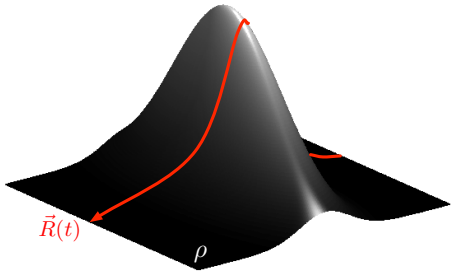


Figure 4.2.1: In 2D transfer function space, peaks along the image of the ray are in general not the same as peak points of the transfer function ρ.

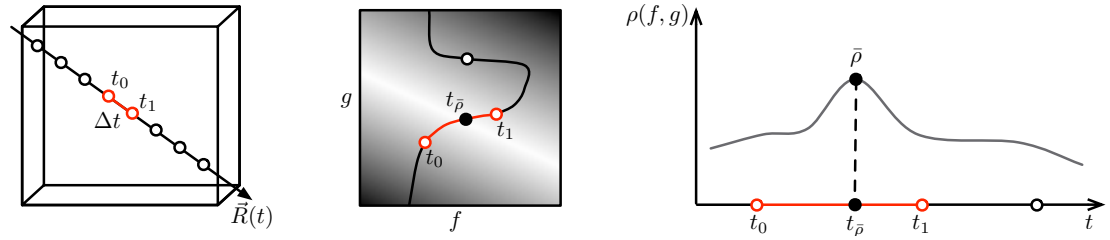
4.2.1 Separable Transfer Functions

When a multidimensional transfer function is separable into 1D transfer functions, one can use 1D peak finding with some modifications. Even when peaks are close together, it is most efficient to assume one peak per ray segment and employ one isosurface solving routine (as opposed to separate solvers for separate dimensions). Instead of using the secant method, bisection is used to determine whether a root for any dimension exists in the left or right half of a bracket. While this can yield shading ambiguities when roots exist in both subfields of $\mathbf{f}$, in practice the quality is good and performance is as fast as 1D peak finding (comparisons will be shown later in Section 4.3.1). Unfortunately, this trivial extension only works for a small subset of multidimensional classifications.

4.2.2 General Multidimensional Transfer Functions

Peak finding in general multidimensional functions requires a different approach. Both enumerating peaks and solving at isosurfaces are difficult (if not intractable) for M-dimensional transfer functions, $M > 1$. Rather than solving for a specific peak value on a segment $[\mathbf{f}_0, \mathbf{f}_1]$ (Figure 4.2.2), note that local maxima can be found dynamically along the ray with sufficient sampling, and that:

- Sampling in transfer function space ρ is less expensive than sampling in world space $\mathbf{f}$.
- Since ρ and $\mathbf{f}$ are compact, a contraction in $\mathbf{f}$ yields a contraction in $\rho(\mathbf{f})$.



(a) A straight ray $\vec{R}(t)$ in world space

(b) The curved path of $\vec{R}(t)$ in transfer function domain, $\mathbf{f}(t) = (f(\vec{R}(t)), g(\vec{R}(t)))$

(c) transfer function profile along the ray, $\rho \circ \mathbf{f}(t)$.

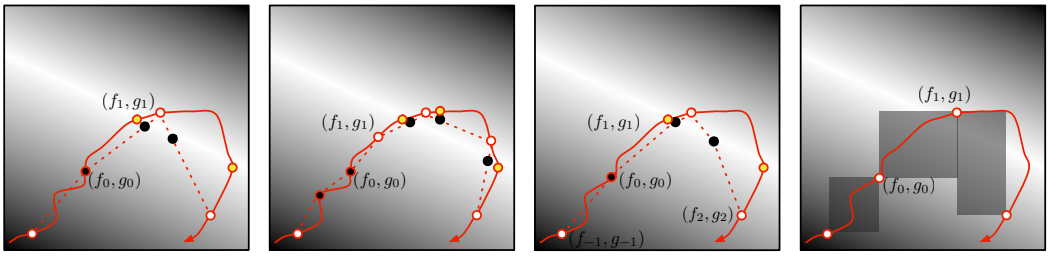
Figure 4.2.2: Classification of multivariate data, and locating peaks between segments.

- Finding the correct local maximum $\overline{\rho}$ on a given $[\mathbf{f}_0, \mathbf{f}_1]$ is more important than accurate world-space location $t_{\overline{\rho}}$.

One is chiefly interested in the intersection of the image of the ray with peak manifolds in transfer function space. Sampling directly along the ray image is costly, as it requires an increase in the world-space sampling rate. However, it is inexpensive to sample along an *approximation* of that image in transfer function space itself. Such an approximation can be parameterized from world-space points $\mathbf{f}_i$, and sampled directly in transfer function space ρ.

As with 1D peak finding, though one must still sample at the Nyquist limit in ρ-space, one only needs to sample $\mathbf{f}$ such that ρ is monotonic on each $[\mathbf{f}_0, \mathbf{f}_1]$ – a less stringent requirement. Thus a parameterization of an approximation of a segment $[\mathbf{f}_0, \mathbf{f}_1]$ in transfer function space is proposed, as illustrated in Figure 4.2.3. As $\Delta t = [t_0, t_1]$ contracts, the segment connecting $\mathbf{f}_0, \mathbf{f}_1$ contracts to approximate $\rho(\mathbf{f})$, as shown in Figure 4.2.3 (a) and (b). One can directly query the transfer function along these segments, identifying a local maximum and using that as a peak. The rest of this section explores chord and spline parameterizations (Figure 4.2.3 (a-c)), using ray marching or scanline sampling. Curve approximations capture classified features better than area elements (Figure 4.2.3(d), [Kra08]), and are less costly when dynamically computing maxima or integrals.

When sampling in transfer function space, time complexity is linear per ray segment in the worst case, as opposed to constant for postclassification, 1D peak finding, and both



(a) chord segments (b) chord segments, $\Delta t/2$ (c) spline segments (d) area

world-space sample (t_i) approximate peak $\tilde{p}$ actual peak $\bar{p}$ peak at sample point

Figure 4.2.3: Parameterizations for sampling in transfer function space $\rho(f,g)$.

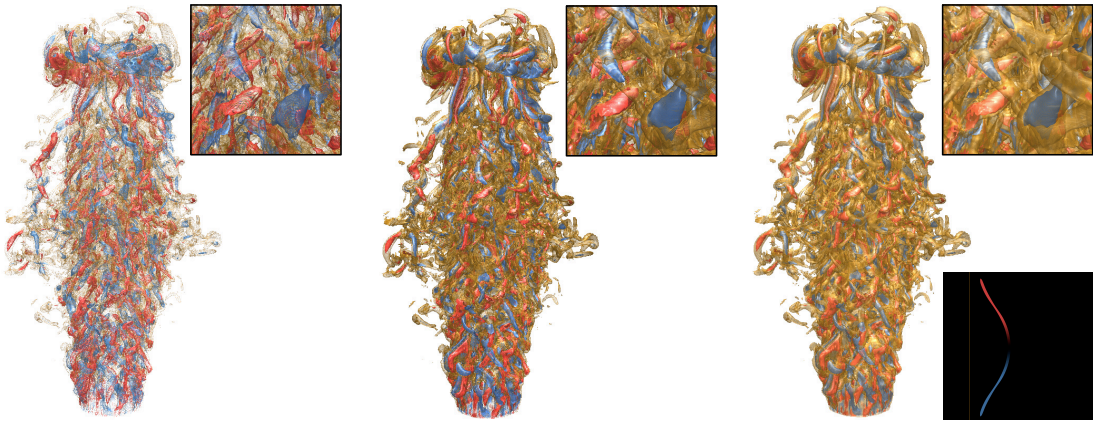
1D and multidimensional preintegration. Unlike with 1D peak finding, one must look for a peak on each segment, since it is not known in advance if it exists in $[\mathbf{f}_0, \mathbf{f}_1]$ or not. However, due to the contractive behavior of $\rho(\mathbf{f})$, few samples in ρ are necessary when samples in $\mathbf{f}$ are close. Assuming ρ is Lipschitz, the number of samples needed on $\rho(\mathbf{f})$ will be bounded, implying average-case constant time complexity per segment. In practice, sampling monotonic regions of ρ incurs small cost. More samples in world-space necessitate fewer samples in transfer space, and visa-versa; one is interested in finding a good equilibrium.

4.2.3 Chord Parameterization

The simplest method of searching in transfer function space is to parameterize the segment between $[\mathbf{f}_0, \mathbf{f}_1] \in \rho \subset \mathbb{R}^M$ as a line, and sample along that chord. Like 1D peak finding and preintegration, this requires fetching front samples and storing back samples $\mathbf{f}(t_1)$ and $\mathbf{f}(t_0)$, respectively. One needs to compute a constant to normalize samples over this segment:

$$\begin{aligned} L_f &= ||(\mathbf{f}_1 - \mathbf{f}_0)|| \\ d_s &= \Delta s \, W / L_f, \end{aligned} \tag{4.2}$$

where Δs is a sampling step in ρ (pixels per sample in transfer function space) and W is the discretization of the transfer function (W=1024 for a $1k^2$ texture).



(a) postclassification $\Delta t = 2$, 5fps (b) postclassification $\Delta t = 0.5$, 0.36fps (c) peak finding (ray march, chord) $\Delta t = 2$, 4.3fps

Figure 4.2.4: Volume rendering of a 2-channel fluid dynamics data set consisting of vorticity magnitude and normalized helicity. The transfer function (shown in the lower-right corner) is chosen to visualize surfaces of medium vorticity (yellow) and high-vorticity regions (red and blue). In the latter, normalized helicity is considered as a secondary variable to color strong vortical regions by direction of rotation. Multidimensional peak finding lets us quickly and accurately render multi-criteria vortex features without explicit mesh extraction.

Then the chord is parameterized as a ray $\mathbf{F}(s)$, where $s \in [0,1]$,

$$\begin{aligned} \mathbf{d}_f &= (\mathbf{f}_1 - \mathbf{f}_0) \\ \mathbf{F}(s) &= \mathbf{f}_0 + s\, d_s \mathbf{d}_f \end{aligned} \tag{4.3}$$

The experiments show that better visuals and performance are achieved with relatively high-resolution transfer functions with smooth (non-pixelated) features and $\Delta s > 1$; $\Delta s = 4, W = 1024$ is used in Figure 4.2.4. For analytical transfer functions such as the one shown in Figure 4.2.5, Δs is set based on the smallest desired feature size. Through this iteration, $s_{\overline{\rho}}$ corresponding to the maximum $\overline{\rho}(\mathbf{F}(s))$ along the segment is found; then the interpolation is performed to find the peak t:

$$t_{\overline{\rho}} = t_0 + \frac{s_{\overline{\rho}}}{t_1 - t_0} \tag{4.4}$$

Having found the peak on this segment, one can proceed to shade (Section 4.2.6).

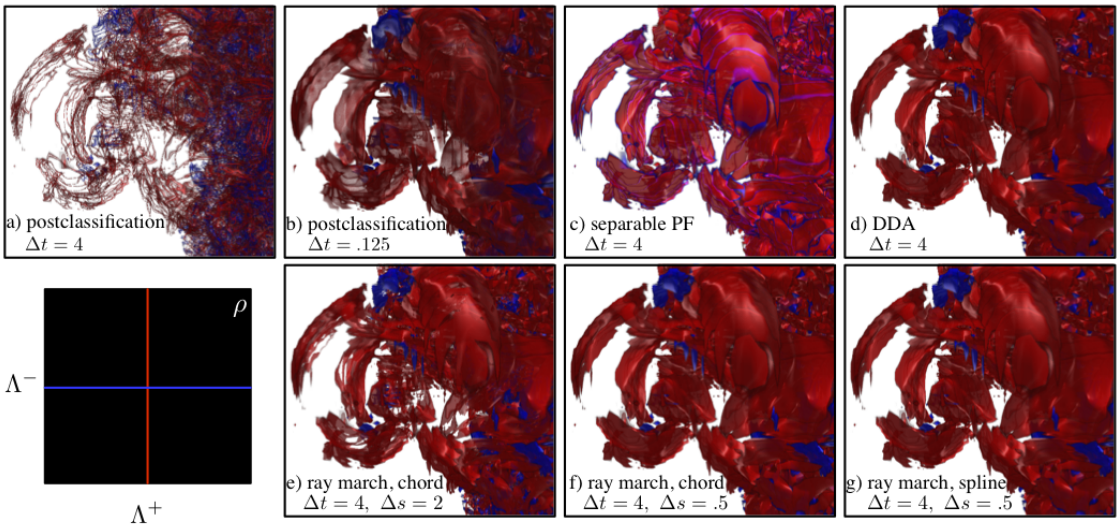


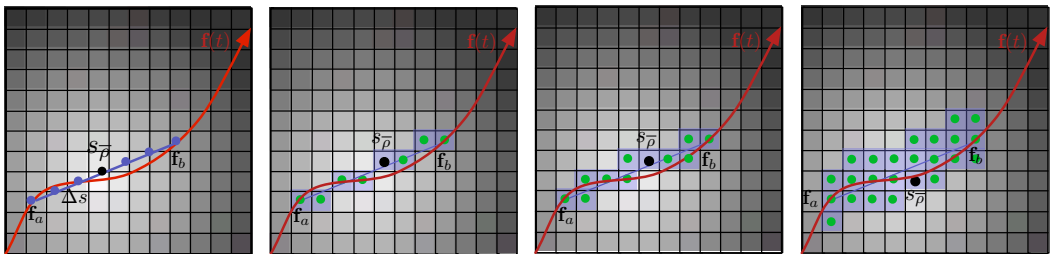
Figure 4.2.5: Comparison of various classification techniques on a close view forward (Λ^+) and backward (Λ^-) FTLE fields of a combustion dataset, classified using a sharp separable Gaussian 2D transfer function, evaluated analytically (sampled at a discretized resolution of 1024^2 in (d)). Rendering of frames (a-g) run at 33, 1.2, 35, 22, 32, 18 and 10 fps, respectively.

4.2.4 Spline Parameterization

Spline interpolation is a logical improvement over piecewise linear parameterization. To accomplish this, one has to maintain a stencil of four world-space samples $\mathbf{f}_{-1}, \mathbf{f}_0, \mathbf{f}_1, \mathbf{f}_2$. A cubic Hermite spline formulation is used, as the four coefficients $h_{ij}(s)$ can be pre-computed and efficiently accessed in a 1D texture on the GPU. As in Section 4.2.3, the chord length L_f is used to parameterize the segment with $s \in [0,1]$ and choose a suitable increment d_s. Although this is an imperfect metric, arc-length parameterization would be too costly. Then the curve is parameterized as a Catmull-Rom spline:

$$\begin{aligned}\mathbf{F}(s) = \mathbf{f}_0 h_{00}(s) + (\mathbf{f}_1 - \mathbf{f}_{-1}) h_{10}(s) \\ + \mathbf{f}_1 h_{01}(s) + (\mathbf{f}_2 - \mathbf{f}_0) h_{11}(s)\end{aligned}$$

Interpolating splines should improve the adherence of the approximating segments to the image $\rho(\mathbf{f}(t))$, providing smoother results with fewer world-space samples. However, the



(a) ray marching (b) DDA (c) Wu (d) slab

Figure 4.2.6: Scanline sampling approaches.

added cost of maintaining a 4-point stencil and evaluating the spline makes this approach impractical for most 2D transfer functions, compared to simpler chordal parameterization with more samples. The technique begins to be useful when the data itself is extremely noisy and world-space samples are farther apart, such as in the example depicted in Figure 4.2.7.

4.2.5 Scanline Sampling

When $\mathbf{f}$ is quantized to 8-bit or lower precision and the user requires more precise classification, it is useful to employ low-resolution (256^2) piecewise-constant transfer functions. To cheaply and accurately find peaks in such functions, a scanline algorithm is employed instead of ray parameterization. A digital differential analyzer (DDA), namely Bresenham's scanline algorithm [Bre65], is used to scan the chord from $\mathbf{f}_0$ to $\mathbf{f}_1$ in discretized $\mathbb{R}^M$-space. This better guarantees that features in ρ will not be missed. 2D DDA is similar to ray marching (Section 4.2.3), except one parameterizes the distance between pixel centers, and march along either the X or Y axis, whichever is greater, incrementing the differential and terminating when one reaches the endpoint on that axis. Instead of the position along the chord, the position along the major axis is used to determine $s_{\overline{\rho}}$ and again interpolate to find $s_{\overline{\rho}}$. This is illustrated in Figure 4.2.6(b).

For comparison other scanline methods were implemented, such as Wu's algorithm [Wu91] (Figure 4.2.6 (c)), which guarantees every pixel between endpoints in ρ will be scanned. However, this approach was slower and did not provide better quality. This is likely because sampling along a chord itself is only an approximation. A modification to the DDA

algorithm was also made to rasterize slabs (Figure 4.2.6 (d)), which produces fewer artifacts but is more expensive, and can overestimate the number of peaks present. Overall, the experiments show that sampling on chords is best for most cases. DDA can be useful for lower-resolution 2D transfer functions, or when the user does not wish to control Δs.

4.2.6 Integration and Shading

Since the peaks are found in between every world-space sample, one does not need to choose a strategy for peak samples with samples from standard (postclassified) DVR integration, as done in 1D peak finding [KHW+09]. $t_{\overline{\rho}}$ is used as the root of the isosurface along the world-space ray, and $\vec{R}(t_{\overline{\rho}})$ as the position at which to shade.

Two principal options exist for choosing the gradient of a multifield volume $\mathbf{f}$ when shading:

- Shading multiple data gradients $\nabla f^0, \nabla f^1$, etc. separately, using multiple central-differences neighbor stencils.
- Computing the gradient $\nabla \rho(\mathbf{f})$ of the transfer function, classifying $\mathbf{f}$ at each point of a single central-difference stencil.

Both approaches are expensive, and are responsible for a significant share of the cost of multifield DVR regardless of whether peak finding is used or not. In the presented examples it was opted for the first approach because $\nabla \mathbf{f}$ tends to exhibit higher frequency than separate individual gradients.

4.2.7 Implementation

All approaches presented were implemented in a GPU shader ray caster written in OpenGL and GLSL. To evaluate baseline performance, this renderer is not heavily optimized; it does not employ methods for multiresolution, empty space culling or adaptive sampling. Indeed, applying such techniques to multifield DVR is nontrivial. Simple methods such as precomputing gradients or adaptive sampling could improve performance; however the presented method opted for the simplicity, flexibility and reproducibility. Note that performance can be greatly improved with such optimizations.

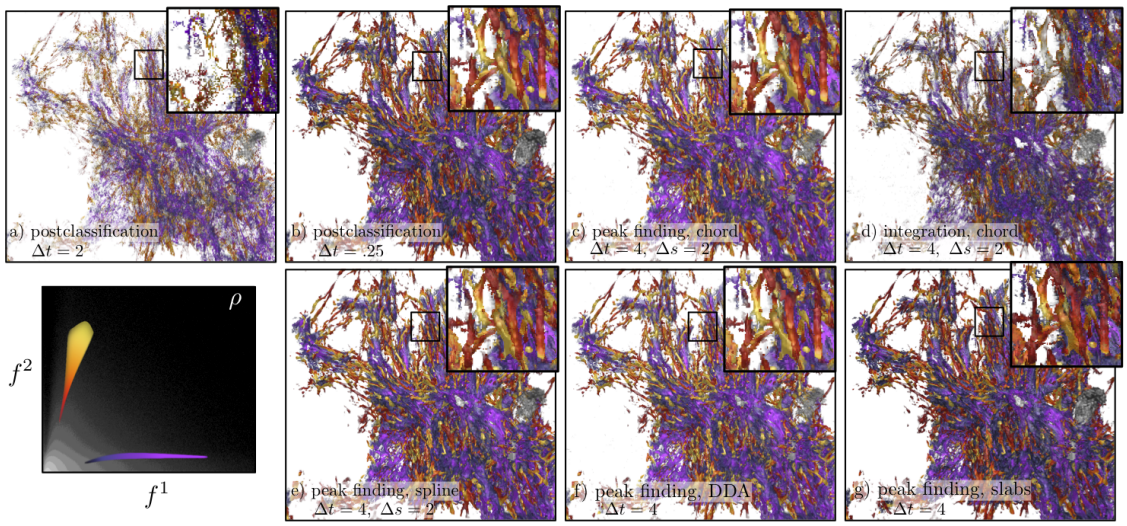


Figure 4.2.7: Classification of matter density (f^1) and dark matter density (f^2) in an Enzo computational astrophysics dataset [NBH+07]. The transfer function highlights ridges in the joint histogram to illustrate where one matter quantity is high relative to the other. A 1024^2 transfer function is used and the rendering at 1024×768 is performed. Frames (a-g) run at 8.5, 1.2, 12.8, 10.6, 7.6, 12.5 and 8.4 fps, respectively.

4.3 Results

Benchmarks were conducted on a 3.0 GHz Intel Xeon and an NVIDIA 285 GTX GPU, at 512^2 screen resolution unless otherwise stated. Rendering high-frequency data and transfer functions requires more samples than rendering low frequency ones, what additionally impacts performance. Generally, peak finding is modestly slower (usually 10-30%) than postclassified ray casting with the same number of world-space samples, but produces results equivalent to sampling at 4x-16x higher rates. In effect, peak finding is an order of magnitude faster for equivalent quality.

Generally, the author recommends using 1024^2 2D textures and the chordal ray marching method with $\Delta t = 2, \Delta s = 2$. There is no major performance difference between peak finding with small and large 2D transfer functions. Aliasing in the transfer function domain is a major source of rendering artifacts; even at 256^2 it is easy to specify features in transfer function space that yield artifacts when undersampled in world space. It is better to use peak finding to improve classification quality and performance, rather than to identify peaks at given pixels in transfer function space. However, multidimensional peak finding

with scanline sampling makes this approach feasible if it is desired. For 256^2 and 512^2 2D transfer functions such as the ones which are used in the gradient magnitude classification examples (Section 4.3.2) the DDA method is slightly faster and better at finding peaks than ray marching. Not needing to control Δs can be seen as an advantage. For analytically constructed transfer functions, it is usually more efficient to use ray marching than sampling into a texture and applying DDA.

4.3.1 Quality Comparison

Figure 4.2.5 shows the comparisons of the results of postclassification, separable peak finding, ray marching and DDA with chordal parameterization, and ray marching with interpolating splines. To compare results a simple Gaussian analytical function $\rho_1(x) = e^{-2^{14}(x-.5)^2}$, yielding a separable 2D function $\rho(u,v) = \sup\{\rho_1(u), \rho_1(v)\}$, is used. Postclassification requires high sampling rates to produce comparable renderings of these features. Separable 1D peak finding is fast, but shows ambiguities where peaks exist in both fields. The true 2D approaches show more accurate results. One can see that even for these sharp features, relatively coarse $\Delta t = 4$ and $\Delta s = 2$ generate good facsimilies. It was observed that further decreasing Δt does not improve quality, but decreases performance. Although this transfer function is deliberately uninteresting, it shows these techniques essentially work, and deliver results comparable to increasing the world-space sampling rate, at significantly lower cost.

Peak finding exhibits similar behavior with non-separable transfer functions, like the one shown in Figure 4.2.4. This figure depicts a two-channel dataset resulting from a fluid flow application. The first variable, vorticity, encodes the magnitude of local rotation of the fluid, whereas the second variable, helicity, describes the alignment of the axis of rotation with the local flow direction. The transfer function shown in the lower right corner is aimed at illustrating vortical motion of the turbulent jet flow described in the simulation data. A vorticity isosurface (yellow) illustrates the larger region of turbulence that is the center of interest in this dataset. To extract individual vortex cores the red and blue lobes of the transfer function capture the rotational direction of the vortices in dependence of the rotational strength. This scenario is a typical representative of multivariable volume rendering in flow analysis applications, where non-separable 2D transfer

functions are employed to illustrate specific features of the flow. In [TGK+04] Tricoche et al. describe the benefits of using volume rendering with this type of transfer functions for flow visualization. In the example depicted in Figure 4.2.4 the high spatial frequency of the data in combination with the chosen transfer function necessitates the use of either very high sampling rates, or the increased fidelity provided by the presented method.

Figure 4.2.7, presents an examination of several approaches from Section 4.2.2 in classifying an Enzo computational astrophysics dataset [NBH+07]. Scientists use joint histograms to understand the statistical relationship between computational variables; multifield volume visualization allows us to show spatial correlation. The transfer function in this example conveys ridges in the joint histogram of density and dark matter density fields, illustrating regions where one quantity is high relative to the other. Note that the transfer function is relatively smooth, though the volume data are high-frequency. In comparing various peak finding modalities, peak finding (c) delivers similar results to postclassification (b) with a 16x higher sampling rate, and at 11x the performance. Image (d) shows the effect of integrating color and opacity along the chord, similar in principle to preintegration. In this example and many others, peak finding delivers results closer to ground truth than preintegrated approaches. The bottom images (e,f,g) show results from non-chordal sampling of the transfer function domain. One can see that interpolating splines (e) provide less aliased results, though with worse performance. DDA sampling (f) appears similar to sampling along the chord both in performance and quality, due to the high resolution (1024^2) of the chosen transfer function. DDA slabs (g) yield even better results, but with some artifacts due to detecting false peaks in the transfer function domain. While spline and slab methods might be appropriate in certain circumstances (such as low-resolution transfer functions) it was found empirically that the simpler chord and DDA approaches yield better results.

4.3.2 2D Gradient Magnitude Classification

One drawback of 1D peak finding is that 1D transfer functions provide limited classification of noisy data from medical and biological sources. 2D functions mapping value and inverse gradient magnitude of univariate data offer better classification of material boundaries. Picking surface features in gradient space is an alternative to isosurfacing;

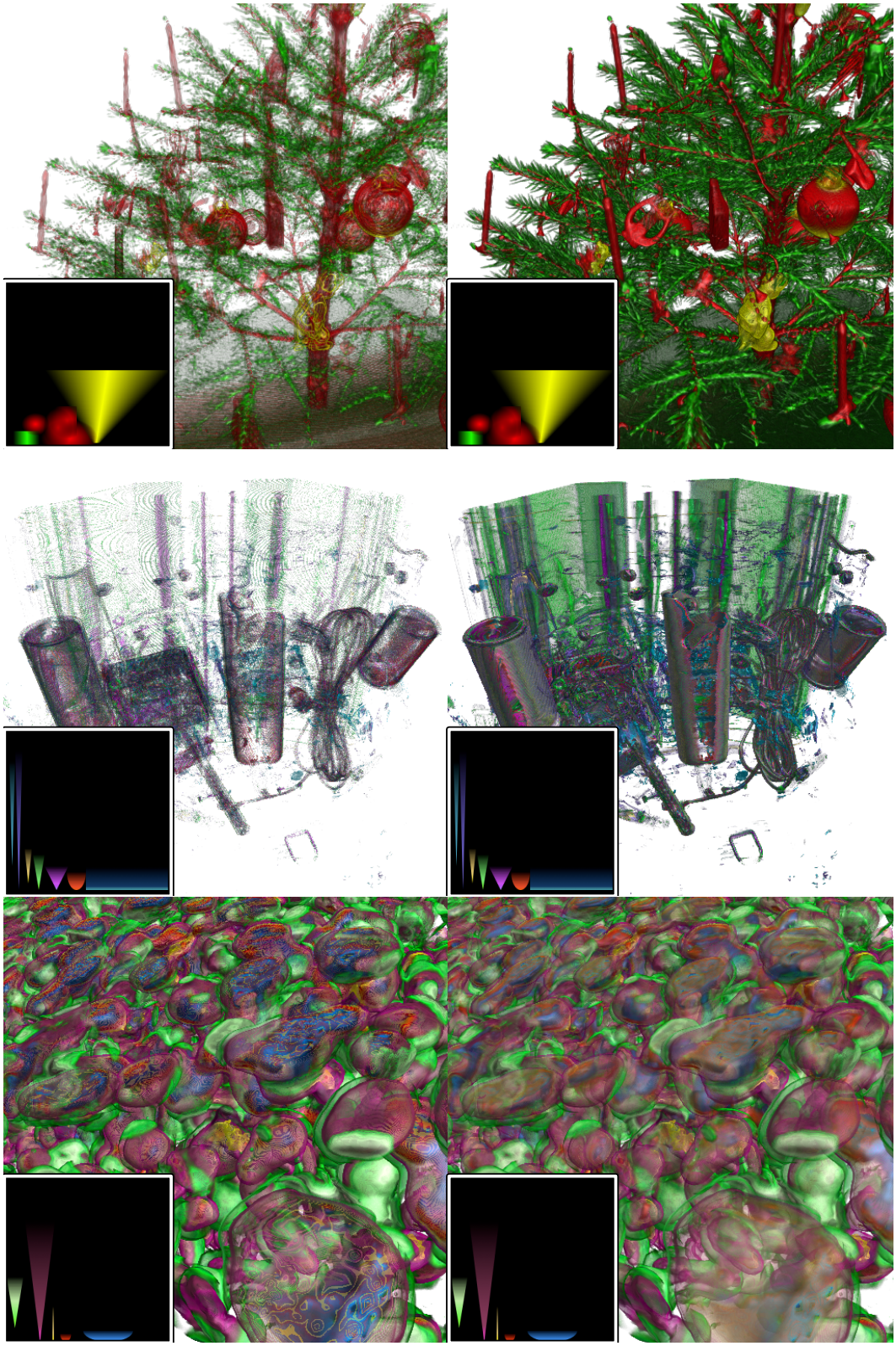


Figure 4.3.1: 2D classifications of value and inverse gradient magnitude, without (left) and with (right) peak finding. From top left to bottom right, these render at 8.0, 7.1, 5.3, 4.0, 12.3 and 9.8 fps, respectively, using scanline DDA sampling.

but peak finding is still useful in its ability to render specified features more accurately at lower sampling cost.

Figure 4.3.1 shows several examples. The *Backpack* and *Christmas tree* are moderate-size Computed tomography (CT) scans with noise rate typical for this kind of acquisition process. With the *Backpack*, peak finding clearly reproduces sharp features in the transfer function that are omitted by standard volume rendering at the same base sampling rate. Even when features are not particularly sharp in 2D TF space, as with the Christmas tree, peak finding frequently allows us to reproduce equivalent quality at a lower base sampling rate and faster overall frame rate. For medical and biological data, 2D peak finding delivers similar advantages as 1D peak finding: namely the ability to isolate and render surfaces from noisy data at higher quality with lower sampling rate.

4.3.3 Higher-Dimensional Multifield Data

Ray marching reduces the search in transfer function space to 1D regardless of the dimension of the classification. This makes it particularly attractive for handling higher-dimensional transfer functions. In Figure 4.3.2, a 4-dimensional CFD combustion simulation [KCH00] is classified, plotting entropy against volume mixture in one 2D transfer function $\nu(f^0, f^1)$, and vorticity and a mixture fraction on another 2D function $\mu(f^2, f^3)$. The transfer functions are mapped on a subset of joint histograms from each data channel. To create a 4D function, ν and μ are convolved using $\rho((f)) = \rho_\alpha(f^0, f^1, f^2, f^3) = \nu\mu$. Then chordal ray marching is used to peak-find directly in this 4D space.

Due to convolution of multiple variables, high frequencies are more common with multidimensional classification. As seen in Figure 4.3.2 (top), standard DVR neglects contributions from sharp isolines, and exhibits noise even at a high sampling rate (8 times the voxel Nyquist limit). Peak finding succeeds in detecting more of these features at the same sampling rate. Though some features may appear to be noise, they do not disappear with a higher peak-finding sampling rate, which indicates that they are actual features specified in the transfer function.

An even stronger argument can be made for peak finding with relatively low-frequency transfer functions in higher dimensions. With convolution of 4 variables, even the large block functions shown in the bottom examples of Figure 4.3.2 can begin to exhibit high

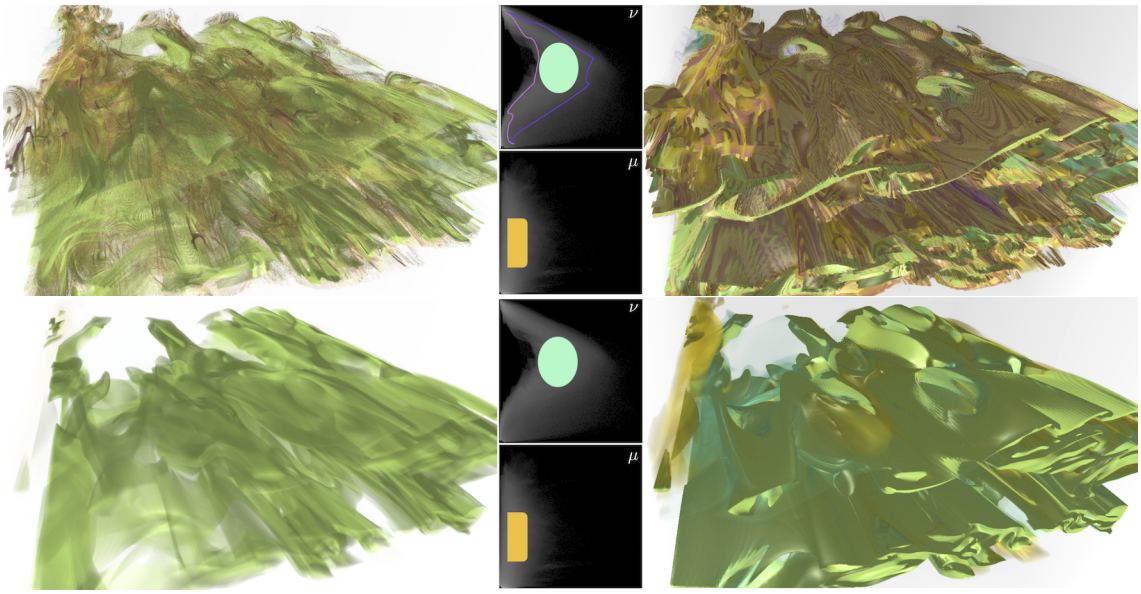
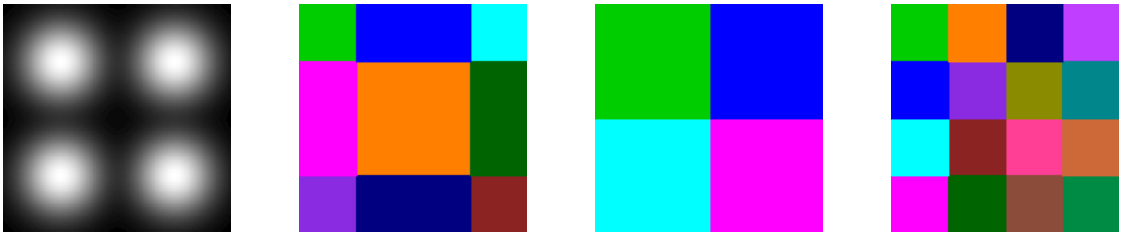


Figure 4.3.2: Visualization of 4-channel combustion simulation data [KCH00] using a 4D transfer function modeled by analytically convolving two 2D transfer function textures ν and μ. Postclassified renderings are on the left and peak finding renderings are on the right. The top row shows a transfer function with peaks, purple and pink isolines. The bottom row shows results with the low-frequency components of the same function. With $\Delta t = 0.5$ for the postclassified examples and $\Delta = 1, \Delta s = 2$ for peak finding with chordal ray marching, these frames render at 0.8, 0.75, 1.6 and 1.6 fps, respectively at 1200×600.

frequencies. Again, these go unnoticed without an explicit algorithm for detecting them, and multidimensional peak finding method excels at reproducing these features.

4.4 Morse-Smale decomposition of the transfer function space

As was shown in the previous section, multidimensional peak finding entails chordal sampling of transfer function space between world-space sampling points. The rendering can be greatly simplified by only performing separable classification in regions that need it. Monotonic regions of transfer function space have no peaks, and can be sampled regularly in world-space with little change in quality. Determining these monotonic regions in 1D is trivial; however in the multivariate case one can make use of topological meth-



(a) A function with four Gaussian peaks

(b) Ascending manifold decomposition

(c) Descending manifold decomposition

(d) Morse-Smale decomposition

Figure 4.4.1: A two-dimensional function with four Gaussian peaks and its decompositions.

ods to analyze the transfer function in the preprocessing stage. This section discusses the application of a topology-based segmentation technique to multidimensional transfer functions.

For the topological analysis of the transfer function space the Morse-Smale complex is used, which gives the information about regions of monotonous gradient behavior. Recall Section 2.1.2.1, where Morse-Smale complex was introduced as an intersection of ascending and descending manifolds of a function. In 2D, the cells of MS complex are regions where the integral lines (gradient lines) have the same origin and destination. It means that the gradient does not change its direction inside of one cell, i.e. local extrema can be found only on the boundary of a cell. Figure 4.4.1 shows a simple function, its ascending/descending manifolds and the MS decomposition. The different regions are color-coded using random colors, as used in the implementation described later in this chapter.

Using the MS decomposition, the computationally expensive multidimensional peak finding can be reduced to samples which fall into different MS cells. Later in this section it will be shown that depending on the topology of the transfer function this can increase the performance of the volume rendering. In any case, the quality of the volume rendering result stays the same as in multidimensional peak finding without topological analysis.

4.4.1 Generating Morse-Smale Complex

To compute the Morse-Smale decomposition of a 2D transfer function domain any of the existing techniques can be used [LLT04, BP07, RGH$^+$10]. The computation of the MS complex is based on the combinatorial gradient field, as defined in Section 2.1.3, which needs to be constructed first. The technique for the computation of such a gradient field proposed by Robins [RWS11] is proven to be correct up to cell complexes of dimension three, i.e. the critical points of the constructed gradient field correspond one-to-one to the changes of the topology of the sublevel sets of the input data, a common definition for critical points in a discrete setting. Therefore the technique was chosen as the first step of the computation. The algorithm considers the lower star of each vertex and constructs the gradient field by simple homotopic expansions. A lower star of a cell x is a set of cells in the neihgborhood of x with the function value lower than $f(x)$. More precisely the lower star $L(x)$ is defined as:

$$L(x) = \{\alpha \in K | x \in \alpha \text{ and } f(x) = \max_{y \in \alpha} f(y)\},$$

where K is a cell complex. Recall from Section 2.1.3 that a combinatorial gradient field on a cell complex consists of pairs of cells (α, β) where α is a face of β. A homotopic expansion is adding to a complex, i.e. "expanding" it, such a pair (α, β) taken from the lower star of each vertex under an assumption that α has no other cofaces in the complex except of β. α is called a free face and (α, β) is called a free pair. The proof of correctness is involved and an interested reader is referred to the original paper for more details on the algorithm.

To extract and simplify the extremal structure of the function and subsequently segment the manifold into Morse-Smale cells the method described in [RGH$^+$10] was used. The simplification is needed to reduce cluttering of critical points in the degenerate regions where the gradient is not defined uniquely. Based on the extremal structure the separate regions are labeled and the pixel values are loaded into a 2D texture, where each label is represented by a unique color.

Morse theory assumes the underlying manifold is continuous, but for many classifications this is not the case. Therefore, in some cases a filter has to be applied to the transfer function image before computing the segmentation. Smoothing filters, such as a discrete

Gaussian or anisotropic diffusion, are primarily used for this purpose. The advantage of using a pre-filtered image instead of the original one is that the discontinuities in the gradient field are smoothed out, which helps to avoid common segmentation artifacts. The filter kernel size and other parameters depend on the transfer function topology and have to be estimated manually. However, one has to be careful with smoothing filtering, sometimes it can result in undesired segmentation when applied strongly, resulting in poor segmentation of the original transfer function and an ineffective querying mechanism for peak finding. Note that smoothing of the transfer function is used only for the purpose of Morse-Smale complex computation. The multidimensional peak finding method still uses the original transfer function for classification. The information from the obtained Morse-Smale complex helps to avoid the computationally expensive peak finding in smooth regions, increasing performance in comparison to the original method.

4.4.2 Modifications to Peak Finding

Applying MS-decomposition to the peak finding algorithm at render time is then simple. At each sample t along the ray, the interpolated two-variate field function $\mathbf{f} = \{f(\vec{R}(t)), g(\vec{R}(t))\}$ is computed. Then the MS decomposition texture described in the previous section is queried at $\mathbf{f}(t)$.

On a ray segment with endpoints t_0, t_1, $\mathbf{f}_0 = \mathbf{f}(t_0)$ and $\mathbf{f}_1 = \mathbf{f}(t_1)$ is computed, and note that:

- if $\mathbf{f}_0 = \mathbf{f}_1$, then the transfer function is monotonic on that segment, and one can omit peak finding.
- if $\mathbf{f}_0 \neq \mathbf{f}_1$, then the transfer function is (potentially) non-monotonic, thus one needs to perform peak finding.

The test is not perfect; it does not consider whether the field functions themselves are monotonic. Indeed, peak finding is often applied in sampling at sub-Nyquist rates, illustrating isosurfaces where they may exist. Here, the goal is not to ensure robustness, but to more aggressively reproduce high-frequency features at lower sampling rates. MS decomposition is a way of determining where such features do not exist, without having to subsample in transfer function space. However, in most cases, the MS-decomposition

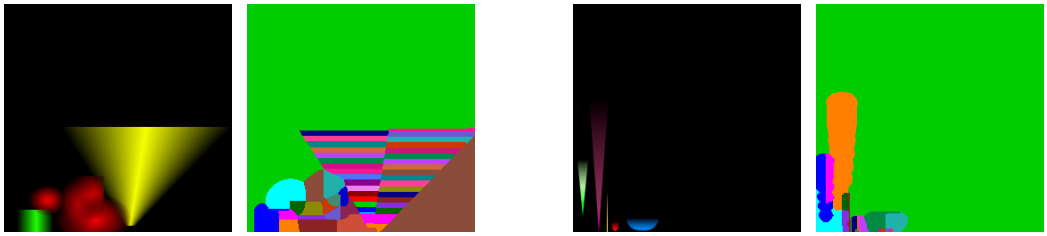


Figure 4.4.2: Transfer functions and their MS decomposition results: Christmas tree (left) and zebrafish embryo (right).

provides an effective query for regions in transfer function that have monotone behavior. By detecting these regions along the ray, one can avoid expensive sampling in transfer function space peak finding.

Peak finding with MS-decomposition can equally be employed in a modality similar to preintegration, where the integral in the transfer domain is computed dynamically as opposed to statically form a lookup table. In practice, peak finding is qualitatively preferable in cases of high-frequency classification, regardless of whether MS-decomposition is used.

4.4.3 Results of applying the Morse-Smale decomposition

The overall goal of MS-decomposition is to perform peak finding only where necessary, and (ideally) to achieve frame rates closer to standard volume rendering at the same world-space sampling rate while peak finding. Peak finding is especially beneficial for classifications with high frequencies (i.e. sharp peaks). At the same time, 2D classification is most commonly applied to univariate volumes with gradient magnitude modality. For the experiments, CT and microscopy data with 2D gradient-magnitude classification and relatively sharp features was chosen.

Note, however, that it is difficult to apply topological analysis to these kinds of functions. A two-dimensional discrete function, like the one shown in Figure 4.4.2, can not be approximated by a continuous Morse function. Large areas of constant function value contradict one of the conditions for a function to be Morse. Moreover, sharp peaks mean, that the function is not C^1 differentiable, in some cases it is not even C^0. Unfortunately,

these properties are typical for transfer functions. This puts some limits on the application of the concepts, which originate from the differential topology. Nonetheless, as a means of partitioning transfer function space into regions of uniform gradient, the MS complex proves useful.

Figure 4.4.2 shows the transfer functions which were used to render the Christmas tree and zebrafish embryo datasets and their MS segmentations. The transfer functions can be generated using any of the available transfer function generation tools. The form of the widgets strongly depends on the data and the corresponding application. The rendering results can be observed in Figure 4.4.3, which illustrates the comparison between the direct volume rendering (DVR), peak finding and peak finding using MS segmentation. All images are generated with equal number of initial samples. Although the DVR method is faster, it misses many of important details in the dataset. The other two methods give approximately the same qualitative result with some performance improvement in the third case. Overall, MS-decomposition allows for peak finding at frame rates nearly as high as postclassified volume rendering, given the same world-space sampling rate Δt. In turn, this allows for higher transfer function space sampling rates Δs to be used at lower total performance penalty.

4.5 Summary

This chapter presented a novel technique for optimizing computation of the rendering integral for volume ray casting. Also an improvement to this method, based on the topological image decomposition, was discussed. Exactly as its one-dimensional predecessor, multidimensional peak finding provides compelling results in the presence of noise or in case of high frequency transfer functions.

This method was applied to 2D gradient-magnitude classification of scalar volume data, and to direct volume rendering of regression-line features in joint histograms of multifield data. Ray marching reduces peak finding in any dimension to a 1D search, making this method applicable to 3D and higher-dimensional classifications. It was demonstrated that this method can be applied to 4D multifield classification, and higher dimensions are

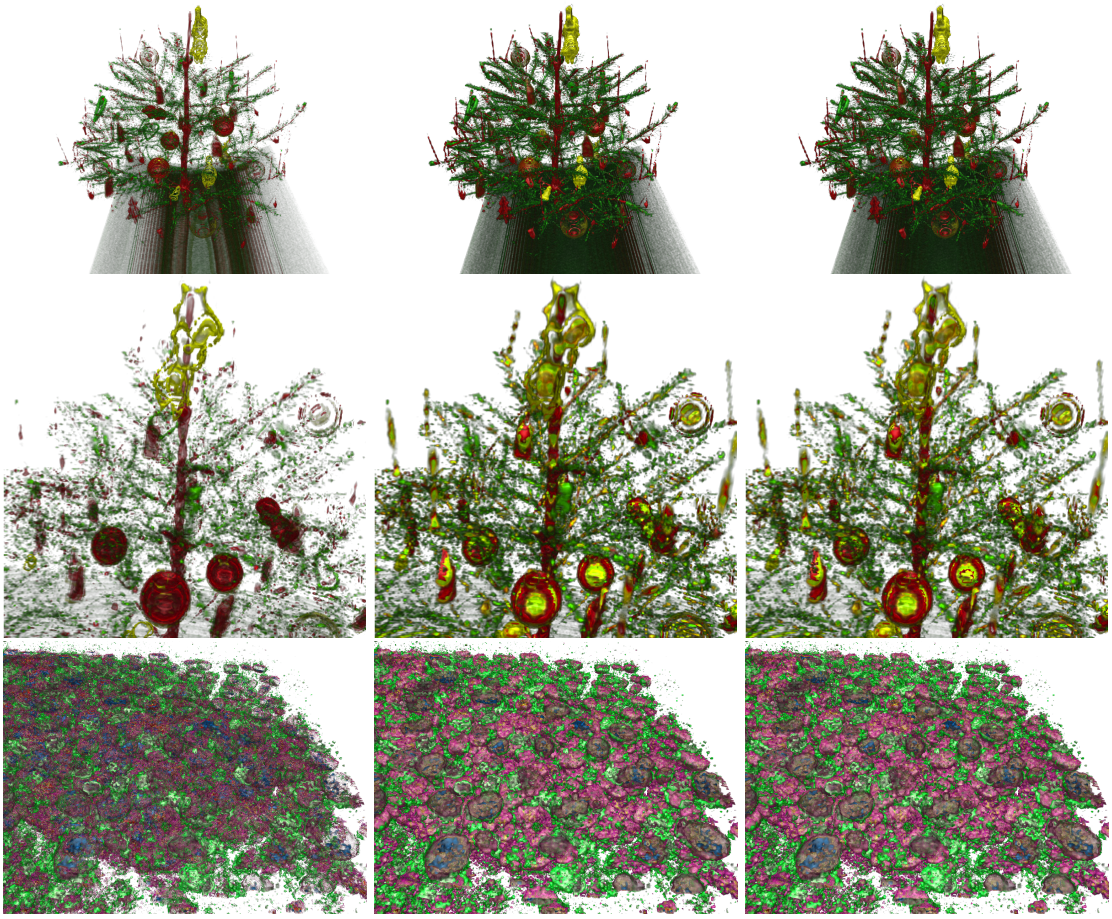


Figure 4.4.3: Method results. Top: Christmas tree dataset. From left to right with standard postclassification (left, 7.0 fps), peak finding (center, 5.5 fps) and peak finding with MS-querying (right, 6.4 fps). Center: closer view of the Christmas tree. Bottom: Zebrafish embryo with performing at 12.5, 8.9 and 10.0 fps, respectively with the above modalities. Renderings captured at 1024x768 resolution on an NVIDIA 285 GTX GPU with 512^2 transfer functions. Renderings using chordal peak finding with a world-space step size of $\Delta t = 2$ and a transfer-space sampling rate of $\Delta s = 2$.

possible. This technique has a potential to be a powerful tool for comparative volume visualization.

The main disadvantage of multidimensional peak finding is that it is not needed if classification is sufficiently smooth. However, as was shown earlier, high frequencies occur even more easily in multidimensional space than in 1D scalar fields. Since isosurfaces cause occlusion, scale-invariant volume rendering with peak finding may not be the best modality for all visualizations. In instances, postclassified rendering could be more useful than the peak-finding. Fortunately, the presented method provides some control over this behavior via the transfer function space sampling rate Δs. Lastly, unlike in 1D, multidimensional peak finding must sample transfer function space between every world-space segment. This is expensive and unnecessary wherever $\mathbf{f}$ is monotonic. However, it does ensure scale-invariance, and in practice the cost of peak finding everywhere is small compared to its benefits.

At the end of this chapter an approach to improve on the performance of the presented method by excluding unnecessary separable classification in regions where it is not needed was discussed. Such regions can be represented by cells of a Morse-Smale complex, where the gradient of a function is monotonic. The Morse-Smale segmentation of the transfer function in the preprocessing stage helps to substitute the separable classification procedure with the two table lookups, giving the opportunity to the performance gain.

This improvement is though limited to 2D transfer functions. While 3D MS decomposition is possible, it would be an expensive preprocess and would consume a large memory footprint. In addition, MS-decomposition is poorly suited for many transfer functions, and parameters for the segmentation and possible pre-filtering have to be specified for each function individually. There are common classes of features having the same range of the parameters, but the automatic detection of these features is not a trivial task. It might, however, give a possible direction for future research. Moreover, one of the advantages of the original multidimensional peak finding method is that it requires no precomputation. Though MS-decomposition provides some speed advantages for common 2D classifications (enabling faster peak finding performance for popular gradient-magnitude classification, for instance), it comes at some precomputation cost.

Multidimensional transfer functions for volume rendering remain an active area of research. Intuitive classifications depend greatly on the data and the application. Usually

a transfer function is drawn by the user and specified as a table of discrete values. The application of continuous MS theory to inherently non-smooth images poses problems. Ironically, high-frequency transfer function for which peak finding is well suited poses the most difficulties for MS decomposition. In the presence of degenerate features, the decomposition can result in oversegmentation with multiple artifacts, the influence of which in the final rendering result highly depends on the case. In the shown experiments, oversegmentation did not reduce the quality of the rendering, but could decrease performance.

Chapter 5

Scale Space based Persistence for the Visualization of two-Dimensional Scalar Fields

With the growing size of the data, feature extraction methods become more and more popular for visual data analysis. Section 2.1.4 described *homological persistence* as an importance measure for critical points, which became a popular tool for feature extraction. However homological persistence does not take into account the spatial extent of a critical point and therefore is not robust against an outlier-like noise.

Lindeberg [Lin94] presented another importance measure for critical points, based on the deep structure of the *scale space*. One can track the critical points of the initial function through multiple scales, analogously to time-tracking of critical points, where the scale parameter is considered to be time t. With increasing scale all critical points merge and then disappear, while the function turns into a constant function. Therefore one can define the importance of a critical point by its life time in the scale space. It is essential to have a very stable tracking of the critical points in scale space for this method to work effectively. When a coarsely sampled data set contains noise the critical lines may be interrupted which severely affects the importance value of the critical points (see Figure 5.3.3e). There are also data sets whose critical points have an infinite lifetime (see Figure 5.3.1b).

This chapter describes a new importance measure for critical points, which builds upon the above two methods. The basic idea is to accumulate the homological persistence value of a critical point through its evolution in the scale space. The new importance measure has the following essential properties:

1. it is able to effectively deal with data containing outliers,

2. it assigns an importance value to each extremal point of the original input data - no preprocessing is necessary,

3. it contains only the sampling of the scale space as computational parameter - all other parts of the algorithm are parameter-free,

4. it is applicable to large data sets due to low memory requirements and practical running times.

5.1 Related Work and Theoretical Background

To develop this method the author took advantage of the knowledge available in different areas of science. The inspiration of the new importance measure is taken from the topological methods for the analysis of scalar fields, especially Morse theory. Because the existing notion of homological persistence had to be extended to deal with a different kind of noise, the use of the scale space theory, known from the area of computer vision, was made to construct a new persistence measure. For the topology computation the state-of-the-art methods are used. This section gives an overview of the theory and work done specifically in these areas.

The following section describes the state of the art and theory related to the scale space area. The usage of this topic is specific for this chapter, therefore it is meaningful to present this information here rather than in Chapter 2. Section 5.1.2 gives a brief review of the Combinatorial Feature Flow fields method which is later used to track critical points in scale space.

5.1.1 Introduction to scale space theory

Scale space is a one-parameter family of functions, obtained by cumulative smoothing of the initial function. Multiple researchers worked independently on the initial concept of the scale space. Recently Weickert et al. [WII99] discovered that the first publication in this area was made by Iijima [Iij62] in 1962. Before this discovery it was commonly believed that Witkin [Wit83] and Koenderink [Koe84] were the pioneers in this topic. The early works of Iijima describe so-called *scale space axioms*. The only smoothing operator which satisfies all of the axioms in $\mathbb{R}^n$ is the convolution with the Gaussian kernel. The scale spaces obtained by the application of this kind of an operator are known as *linear scale spaces*. Another kind of scale spaces, which draws much attention in the recent years, is known as α *scale spaces*. It was originally proposed by Pauwels et al. [PVGFM95] and got deeply investigated by Duits et al. [DFDGTHR04], who has proven that α scale spaces satisfy all basic scale space axioms. For some applications, *non-linear scale spaces* [PM90] might be an interesting solution. For the particular purpose of this chapter linear (or Gaussian) scale space has proven sufficient. Koenderink [Koe84] proposed to investigate the evolution of critical points of the initial function through its scale space, called *deep structure*, see Lindeberg [Lin94] for an extensive introduction. In contrast to the presented approach, the deep structure is usually defined in a continuous setting and is extracted using numerical methods.

Let $f : \Omega \to \mathbb{R}$ with $\Omega \subseteq \mathbb{R}^2$ denote a function that represents the input data. The scale space associated with f is a function

$$L : \Omega \times \mathbb{R}^+ \to \mathbb{R} \tag{5.1}$$

L represents f at various scales. In classical scale space theory one considers $\Omega = \mathbb{R}^2$. Then the linear scale space at scale t is given by convolving the function f with a Gaussian kernel with variance t:

$$g(x,y;t) = \frac{1}{2\pi t} e^{-(x^2+y^2)/2t}; \quad L(\cdot,\cdot;t) = g(\cdot,\cdot;t) * f(\cdot,\cdot).$$

In theory it is usually assumed that the scale space is given on an unbounded domain. This assumption does not hold in reality, because in practice, the domain Ω always has a non-empty boundary $\partial\Omega$. A common case is $\Omega = [0,a] \times [0,b]$.

There are two common ways to define the scale space in this setting. The first one is to extend the function to the whole of $\mathbb{R}^2$, e.g. by extending the function with zero outside its domain.

The second approach is to define the scale space as the solution of the heat equation with Neumann boundary conditions:

$$\begin{aligned} \frac{\partial L}{\partial t} &= \Delta L \quad \text{in } \Omega \times \mathbb{R}^+ \quad \text{(Evolution equation)}, \\ \frac{\partial L}{\partial \nu} &= 0 \quad \text{in } \partial\Omega \times \mathbb{R}^+ \quad \text{(Boundary condition)}, \\ L &= f \quad \text{in } \Omega \times \{0\} \quad \text{(Initial condition)}, \end{aligned} \tag{5.2}$$

where ΔL denotes the spatial Laplacian of L and ν denotes the outer-unit-normal of Ω. The latter approach is used to define the scale space for a bounded domain. Duits et al. [DFFP03] compared these two approaches and provided some mathematical justifications for enforcing Neumann boundary conditions. If one represents a bounded 2D scalar field as a grey valued image, then using this kind of boundary conditions guarantees that no grey value flux goes through the boundary. It means that the values inside of the domain are not influenced by the artificial values outside of it, moreover the average grey value is maintained, what is especially important at a large scale. With this kind of boundary conditions the given 2D function on a rectangular bounded domain approaches the behavior of an unbounded smooth function, for which the classic scale space theory is defined. If the scale s approaches infinity, the function tends to a constant function $f_s(x,y) = C$, where C is the average value of the original function.

An illustration of the scale space of a simple 1D function defined on a bounded domain is shown in Figure 5.1.1.

5.1.2 Combinatorial feature flow fields

The idea behind the scale space persistence measure is based on the evolution of the homological persistence value of a critical point in the scale space of the given function. To do this, the scale space $L : \Omega \times \mathbb{R}^+ \to \mathbb{R}$ corresponding to f is represented as a time-dependent function.

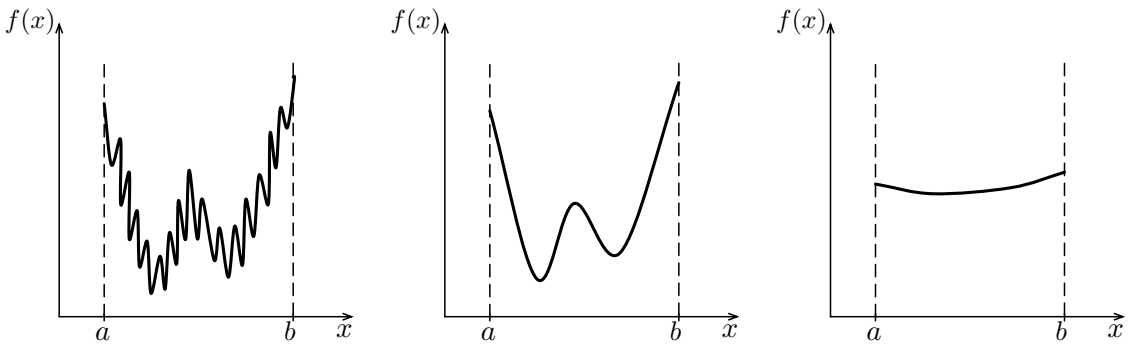


Figure 5.1.1: A 1D function $f(x)$ represented at different scales in scale space: a) $f(x)$ at the initial scale $s_0 = 0$, b) $f(x)$ at an intermediate scale $s_m > s_0$, c) $f(x)$ at a large scale $s_n > s_m$.

Combinatorial feature flow fields (CFFF) [RKWH11] is the method to track critical points of such time-dependent data in the context of discrete Morse theory. The benefit of this approach is that one can incorporate homological persistence straightforwardly. The idea of this method was first proposed by King et al. [KKM05] and later the computation algorithm was developed by Reininghaus et al. [RKWH11].

For better understanding first a basic intuition for CFFF for minima has to be given. To do this, one needs the notion of a basin of a minimum c. This is a subset of the domain that is given by all points that are transported to c by the flow of the gradient vector field (see Figure 5.1.2 for a one-dimensional illustration). Two minima c and d of two subsequent time steps are then tracked if and only if they are contained in each others basin. In the combinatorial setting this can be stated formally as follows.

Let $(V_\ell)_{\ell=0,1,2,\ldots}$, where ℓ stands for subsequent time stamps, denote a sequence of combinatorial gradient fields, as presented in Section 2.1.3, defined on a cell graph $G = (N, E)$. To simplify notation only the description of the tracking of minima is given here. The tracking of maxima is done analogously. Let F_ℓ denote the stationary combinatorial flow map of slice ℓ induced by V_ℓ that maps a 0-cell to the end point of its uniquely defined combinatorial 0-streamline. Let $W_\ell \subseteq N$ denote the fix points of the map $F_\ell \circ F_{\ell+1}$. The critical minima lines that pass through the time steps ℓ and $\ell+1$ can now be defined as the set of pairs $(u, F_{\ell+1}(u))$ with $u \in W_\ell$. The critical lines of the whole sequence are then given by concatenating these segments. An illustration of these basic concepts is given in Figure 5.1.2.

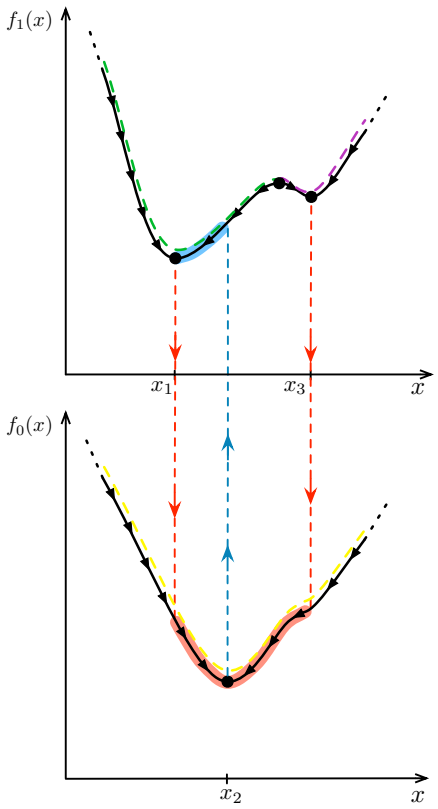


Figure 5.1.2: A 1D function with two time steps depicting the basic concepts of combinatorial feature flow fields. The flow maps F_0 (red) and F_1 (blue) induce the minima pairs (x_1,x_2), (x_3,x_2) and (x_2,x_1). Therefore, the pair (x_1,x_2) builds a segment of a critical minima line. Green, purple and yellow dashed lines show the basins of minima x_1, x_2 and x_3.

Note that maxima can be tracked similarly, while saddle critical lines are a little more intricate to compute. An importance measure of a single critical line, called integrated persistence, can be defined by the sum of the homological persistence values of its nodes.

5.2 The method

This section describes the algorithmic pipeline for the computation of the scale space persistence measure for minima and maxima in 2D scalar fields. The computation that yields the measure consists of five steps which are depicted in Figure 5.2.1.

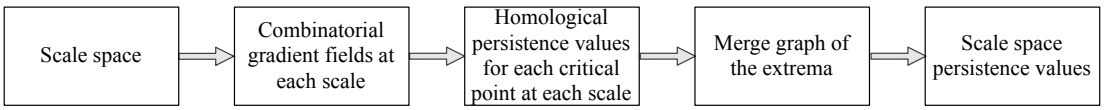


Figure 5.2.1: Five steps needed for the computation of the scale space persistence values.

For the first three steps existing techniques are used and therefore only provide a brief description is provided. An interested reader can refer to the corresponding publications if more information is desired. The computation of the scale space persistence values based on the merge graph of the extrema is explained in more detail. While the presented importance measure is in principal well defined for general domains, the algorithmic pipeline is restricted to data f defined on a rectangular 2D domain $\Omega = [0,a] \times [0,b]$ and sampled on a regular grid. Common examples of such data are 2D images.

5.2.1 Scale Space computation

To compute the scale space $L : \Omega \times \mathbb{R}^+ \to \mathbb{R}$ of a function $f : \Omega \to \mathbb{R}$ one needs to solve the heat equation with Neumann boundary conditions (5.2). For the common case of a rectangular 2D domain there is a closed form solution as presented in [DFFP03]

$$L(x,t) = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \left(\int_{\Omega} f \, l_{m,n} \, dx \right) l_{m,n}(x) \, e^{-\left(\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2 \right) t}, \tag{5.3}$$

where the value of $l_{m,n} : \mathbb{R}^2 \to \mathbb{R}$ at $x = (x_1, x_2)$ is defined by

$$l_{m,n}(x) = \frac{2 \cos\left(\frac{m\pi x_1}{a}\right) \cos\left(\frac{n\pi x_2}{b}\right)}{\sqrt{ab}}.$$

Equation (5.3) shows that the scale space of an image can be evaluated exactly in nearly linear time using the efficient fast discrete cosine transform algorithm [CL91]. Since a finite representation of L is required $\mathbb{R}^+$ is restricted to the interval $[0,T]$ which is exponentially discretized using n samples. This exponential sampling of the scale space is a standard approach in the literature [Lin94] since it represents small scale structures well. If all scales of the data should be considered, one can choose T such that $L(\cdot,T)$ is nearly

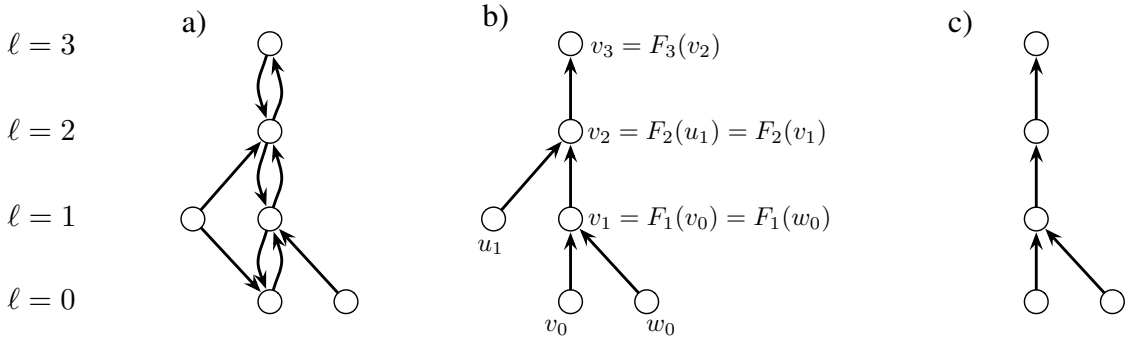


Figure 5.2.2: Constructing the pruned merge graph: a) The initial merge-split graph H, b) the corresponding merge subgraph, c) the pruned merge graph M.

constant. If n is chosen too small, the critical point tracking may be strongly affected since the time steps would change too rapidly. One can also pick a target precision $\sqrt{\varepsilon}$, and determine n automatically as follows.

Let L^k denote the scale space with $n = k$ samples. Then

$$\int_0^T \int_\Omega \left(L^{2k} - L^k\right)^2 dx\,dt$$

is computed for some small value of k. If this value is smaller than ε one can stop the computation and use L^{2k} for the further steps. Otherwise one doubles k. Note that the individual samples of the scale space can be computed independently using the fast discrete cosine transform algorithm. It therefore suffices to compute only the samples of L^{2k} which are not already contained in L^k.

5.2.2 Combinatorial Gradient Fields

The next step is the computation of a sequence of combinatorial gradient fields (V_ℓ) that represents the n slices of the scale space L. To efficiently compute V_ℓ one can make use of an algorithm proposed by Robins [RWS11]. The same algorithm was used in Section 4.4.1 of this thesis. Recall that the algorithm considers the lower star of each vertex for construction of V_ℓ and since the decomposition of a grid into the lower stars of a function is disjoint, the computation of V_ℓ can be done in parallel. The parallel version of the algorithm was implemented using OpenMP.

5.2.3 Homological Persistence

For the computation of the scale space persistence one needs the homological persistence values, which have to be calculated for critical points of each slice of the scale space. A simple algorithm to compute the persistence pairs is presented in [CSEM06]. Loosely speaking, this algorithm applies a simple matrix reduction algorithm to a data induced permutation of the boundary operators of the cell complex. Note that this algorithm considers originally all cells of the complex. As shown in [RWS11] the running time of this algorithm can be significantly improved by making use of the Morse-Smale complex induced by V_ℓ.

5.2.4 Merge Graph

Using (V_ℓ) one can now track the minima and maxima of f through the scale space. One approach would consist of a straightforward application of the combinatorial feature flow field method [RKWH11] to extract the critical lines defined by (V_ℓ). Since critical points can appear and disappear in the scale space the extracted critical lines may be interrupted. This has a strong effect on the critical lines beginning in V_0.

Therefore the method presented in this chapter adopts a different point of view regarding the evolution of critical points. One can think of a minimum u in V_ℓ as a representative of its basin. The basin of u is given by the preimage of u under the flow map $F_\ell^{-1}(u)$. Note that in contrast to minima, basins can merge. One can thereby define the notion of merging minima: when a minimum u disappears in the scale space and its associated basin has merged with the basin of another minimum v, one says that u and v have merged. By duality, the same argument applies of course also to maxima and to split events.

Now the definition the directed merge-split graph H can be made. The nodes of this graph consist of all critical points contained in (V_ℓ). The directed edges are given by all pairs $(u, F_{\ell+1}(u))$ and $(u, F_{\ell-1}(u))$ with u being a critical point in V_ℓ (see Figure 5.2.2a).

While critical points may appear in the evolution of a 2D scale space [LP90], they can be regarded as artificially created. Therefore only the merge graph that consists of the subgraph of H containing only the edges $(u, F_{\ell+1}(u))$ (see Figure 5.2.2b) can be considered. The nodes of the graph that are not related to the nodes of V_0 can be removed. To do this,

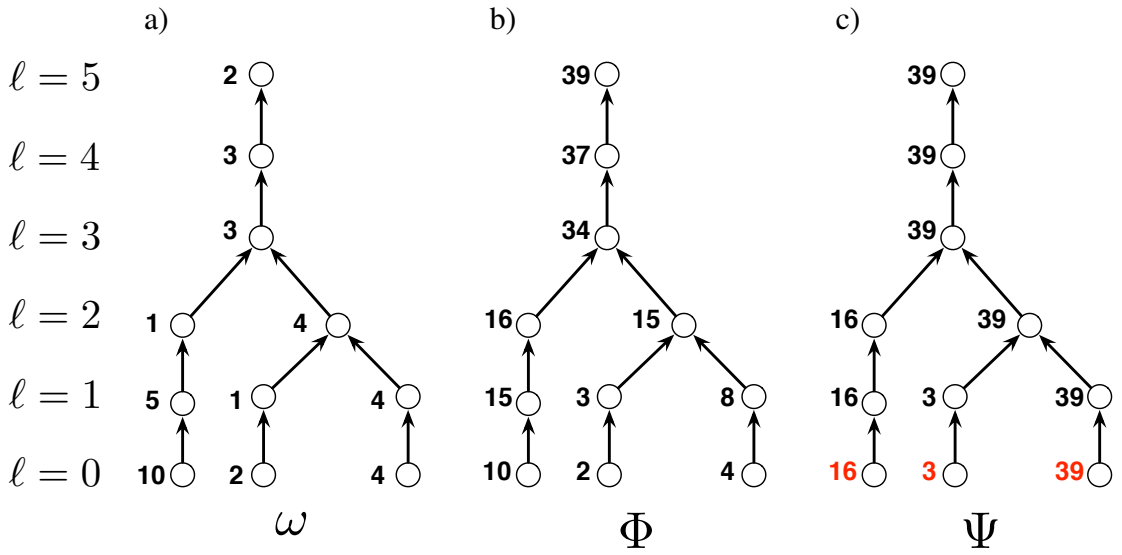


Figure 5.2.3: Scale space persistence computation: a) the merge graph with the node values given by homological persistence ω, the persistence value indicate the presence of an outlier (the left branch of the graph); b) computation of the forward persistence Φ, c) computation of the backward persistence Ψ, the scale space persistence measure (red) is the restriction of Ψ onto the critical points at the finest scale $\ell = 0$. Due to the usage of the persistence value in the back propagation, the outlier on the left is assigned with a smaller value than the dominant critical point on the right.

a breadth first search starting at all critical points of V_0 is applied and all unvisited nodes of M are deleted. This results in the *pruned merge graph* $M = (P,K)$ of the scale space L (see Figure 5.2.2c).

5.2.5 Scale Space Persistence

In Section 5.2.3 persistence values for all critical points in (V_ℓ) were computed. Therefore one can assign these node values $\omega : P \to \mathbb{R}^+$ to the merge graph $M = (P,K)$ (see Figure 5.2.3a). Let $P_\ell \subset P$ denote the nodes in the graph that correspond to the critical points of V_ℓ. The *scale space persistence* of the critical points P_0 of V_0 can now be defined by propagating the information contained in the node weighted graph M. This is done in two steps.

In the first step a derived importance measure $\Phi : P \to \mathbb{R}^+$ called forward persistence is defined. The main idea is to accumulate the persistence values ω from P_0 through to

P_{n-1}. The forward persistence Φ of a critical point u at some scale $\ell+1$ is the sum of the homological persistence value of u and the Φ values of all neighbors of u located at the previous scale ℓ (see Figure 5.2.3b). Formally, Φ is defined iteratively by:

$$\begin{aligned} \Phi|_{P_0} &= \omega|_{P_0}, \\ \Phi|_{P_{\ell+1}}(u) &= \omega(u) + \sum_{q \in Q_u} \Phi|_{P_\ell}(q) \end{aligned} \tag{5.4}$$

with $Q_u = \{q \in P \mid \exists (q,u) \in K\}$.

In the second step Φ is used to define the scale space importance measure. Therefore, one needs to propagate the forward persistence values Φ back to P_0. To do this node values Ψ, called backward persistence, are computed. On the coarsest scale $\ell = n-1$, Ψ is equal to Φ. If a critical point u at scale level ℓ has multiple predecessors at the finer level $\ell-1$, the most important predecessor with respect to homological persistence ω is assigned the value $\Psi(u)$. The others are assigned with their forward persistence values Φ. Formally, Ψ is defined iteratively by:

$$\begin{aligned} \Psi|_{P_{n-1}} &= \Phi|_{P_{n-1}}, \\ \Psi|_{P_{\ell-1}}(u) &= \begin{cases} \Psi|_{P_\ell}(v) & \text{if } u = \underset{q \in Q_v}{\arg\max}\, \omega(q) \text{ with } (u,v) \in K \\ \Phi(u) & \text{else} \end{cases} \end{aligned} \tag{5.5}$$

The novel importance measure *scale space persistence* is now simply given by $\Psi|_{P_0}$. Note that the node v that appears on the right hand side of equation (5.5) is uniquely defined since K does not contain cycles.

A simple example for the computation of Ψ is shown in Figure 5.2.3c. The fact that the values of Φ are back propagated based on the information contained in ω can be seen between $\ell = 2$ and $\ell = 3$.

5.3 Results and Discussion

The next section contains some results of the presented method applied to a synthetic test data set and a real-world data set. In both cases the new importance measure is compared

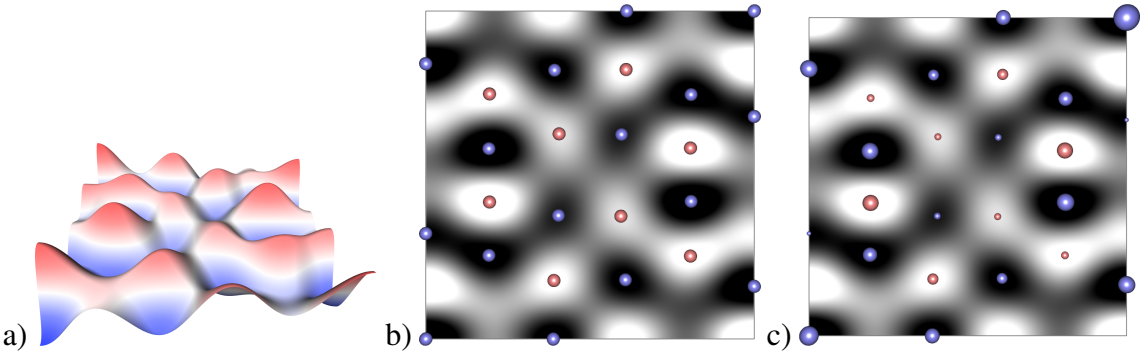


Figure 5.3.1: A synthetic test data set f with $-\Delta f = \lambda f$: b) red and blue spheres depicting the maxima and minima of f scaled by scale space lifetime, c) red and blue spheres depicting the maxima and minima of f scaled by scale space persistence. Since f is an eigenfunction of the Laplace operator, the scale space lifetime is infinite for all critical points. Scale space persistence however, assigns a sensible importance measure to the critical points of f.

to the classical homological persistence measure and the scale space lifetime measure. Finally, the performance and scalability of the method are evaluated.

5.3.1 Comparison to Scale Space Lifetime

Using scale space lifetime as an importance measure can be problematic when the critical points of the given function f live extremely long in its scale space L. It can even happen that all critical points have an infinite lifetime, making it impossible to differentiate between them using their lifetime. This happens for example when f is an eigenfunction of the Laplace operator, i.e. $-\Delta f = \lambda f$, and fulfills the Neumann boundary condition (5.2). The scale space L of such a function is then given by $L(x,t) = f(x)\,e^{-\lambda t}$. This implies that if f has a critical point at x_0, then $L(x_0,t)$ is a critical point for all t. The scale space lifetime of each critical point of f is therefore infinite. In contrast, scale space persistence is able to differentiate between the critical points of f. This can be illustrated with the eigenfunction

$$f(x) = \cos(7x_1)\cos(x_2) + \cos(5x_1)\cos(5x_2),$$

with $x = (x_1, x_2)$ and $\Omega = [0,\pi]^2$. Figure 5.3.1 shows a depiction of f, and a comparison of scale space lifetime with scale space persistence.

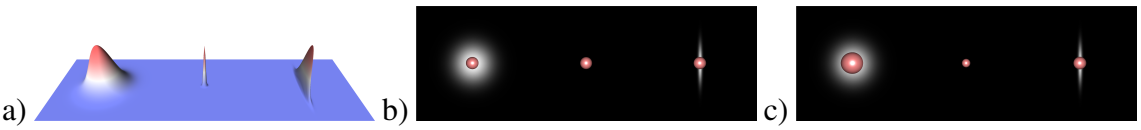


Figure 5.3.2: A synthetic test data set f containing three maxima: a) a height field visualization of f, b) red spheres depicting the maxima of f scaled by persistence, c) red spheres depicting the maxima of f scaled by scale space persistence. While classical persistence assigns the same importance to all three maxima, scale space persistence takes the spatial extent of the maxima into account.

5.3.2 Comparison to Homological Persistence

The homological persistence of a function f is invariant under homeomorphisms. In terms of persistence, the importance of a critical point does therefore not depend on its spatial extent. In contrast, scale space persistence takes the spatial extent of a critical point into account. A simple example that illustrates this difference is shown in Figure 5.3.2. This function contains a blob-like, an outlier-like, and a ridge-like maximum. While persistence assigns the same importance value to all maxima, scale space based persistence assigns different values to these maxima.

5.3.3 Noise Robustness

To test the robustness of the new importance measure with respect to noise a synthetic data set was produced. For this purtope an analytic function on a regular grid of dimension 256×256 was sampled and uniform noise and outlier-like noise was then added to it. This resulted in the function visualized by a height field in Figure 5.3.3a. Since the scale space lifetime measure is very sensitive to a correct tracking of the critical points the scale space had to be sampled densely with $n = 768$. An illustration of L is shown in Figure 5.3.3b, while the pruned merge graph M of the minima of L, with thickness given by homological persistence, is shown in Figure 5.3.3c.

The bottom row of Figure 5.3.3 shows the 38 most important minima and maxima with respect to homological persistence, scale space lifetime, and the new importance measure scale space persistence. Due to the presence of outliers in the data, the homological persistence measure fails to identify a lot of dominant extremal points. While the scale

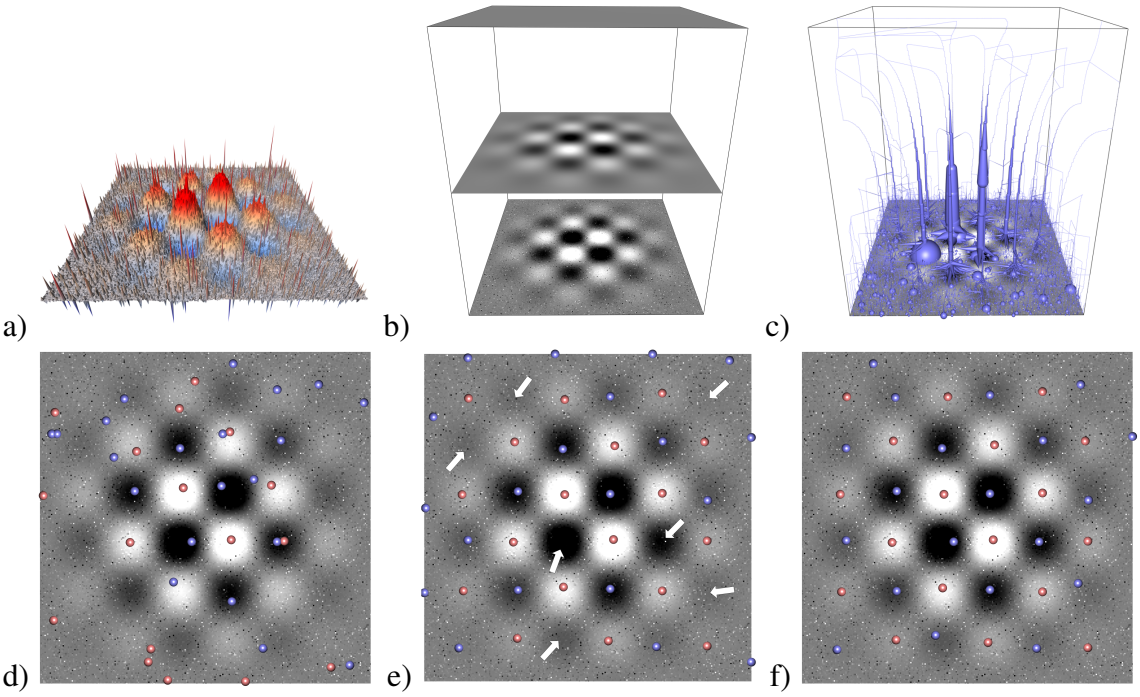


Figure 5.3.3: A synthetic test data set f containing uniform and outlier-like noise: a) a height field visualization of f, b) the scale space L of f, c) the pruned merge graph of the minima of L, with thickness given by homological persistence, d)-f) the 38 most important minima and maxima of f as specified by d) persistence, e) scale space lifetime, f) scale space persistence.

space lifetime measure is able to extract most of the dominant extremal points, it fails to extract some clearly visible ones indicated by the white arrows. This problem stems from the fact that some critical lines are interrupted due to the high amount of noise present at the fine scales of L.

In contrast to these existing importance measures, scale space persistence is able to recover all dominant extremal points present in the unperturbed function.

5.3.4 Elevation Map of a Region on Mars

The method was applied to elevation data of a region on Mars shown in Figure 5.3.4a. This data set was acquired by the Mars Orbiter Laser Alimeter (MOLA) and is provided by NASA (http://pds-geosciences.wustl.edu/) and has a resolution of 128 pixels per degree.

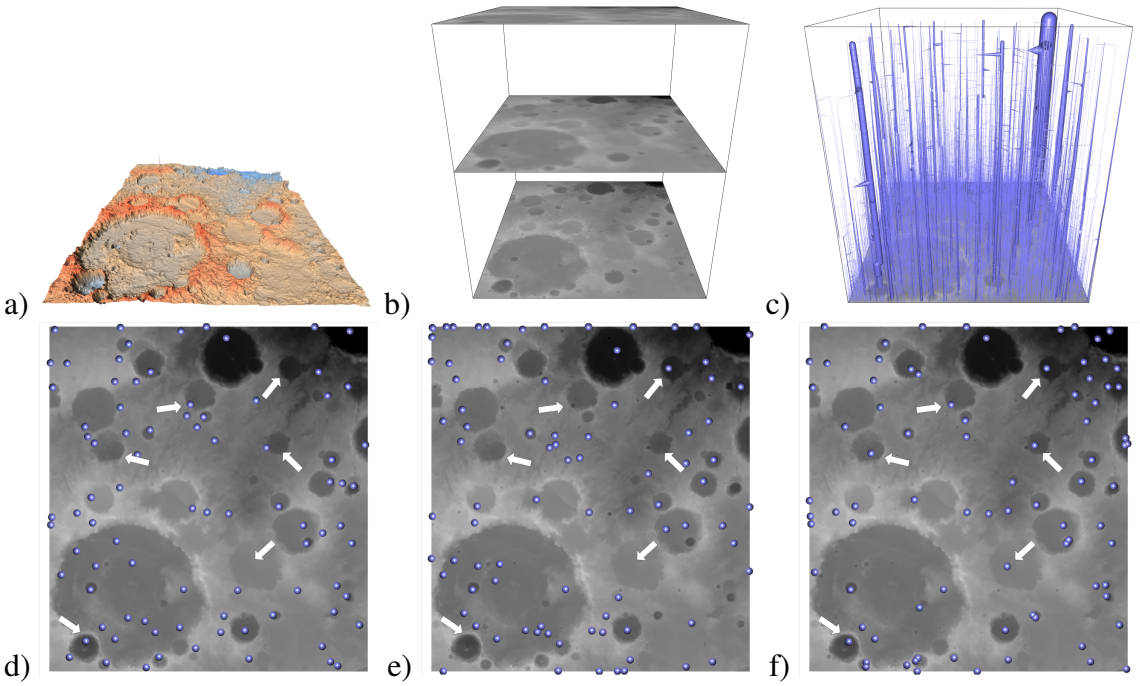


Figure 5.3.4: A real-world test data set f given by the elevation of a region on Mars: a) a height field visualization of f, b) the scale space L of f, c) the pruned merge graph of the minima of L, with thickness given by homological persistence, d)-f) the 85 most important minima of f as specified by d) persistence, e) scale space lifetime, f) scale space persistence.

The selected region has dimensions of 1372×1483. In this kind of data, the domain scientists are interested in the extraction of craters since one can infer many properties of celestial bodies using this information. The craters are represented by minima of the elevation function that have a large extent. Since the presented method takes into account the spatial extent of the extremal points it is well suited for this kind of application.

To enable a fair comparison of the scale space persistence method with the scale space lifetime measure, the scale space of this function had to be sampled very densely with $n = 1024$. The bottom row of Figure 5.3.4 shows the 85 most important minima with respect to homological persistence, scale space lifetime, and the importance measure scale space persistence. The white arrows indicate some of the differences of the three methods. For example, there is a very large but shallow crater at the top-left of the lower-right quadrant indicated by a white arrow. This crater is missed by homological persistence (too

shallow for persistence) and by scale space lifetime (unstable tracking due to sampling). In contrast, the new importance measure assigns a high value to this minimum, since it accumulates the low persistence value over a long time.

5.3.5 Performance analysis

Dataset	Spatial res.	Scale space res. (n)	5.2.1	5.2.2	5.2.3	5.2.4	5.2.5	Total time
Synthetic	256×256	768	51s	154s	309s	85s	1s	10min
Mars	1372×1483	1024	1014s	3072s	6723s	859s	32s	3h 15min

Table 5.1: Performance measurements of the presented method. The columns 4-8 show the running time of the individual steps of the algorithm as described in Section 5.2.

The performance and scalability of the algorithmic pipeline was measured on the two test data sets shown earlier. Table 5.1 shows the running times of the individual steps described in Section 5.2. The running times were measured on a computer using four AMD Operon 6174 CPUs. Since the presented algorithm lends itself to a streaming computation one only needs to have two slices of the scale space in memory at any given time. The memory overhead of the algorithm is therefore negligible.

As can be seen in Table 5.1, most computation time is spent calculating the combinatorial gradient fields using the algorithm described in [RWS11] and the homological persistence values using the algorithm described in [CSEM06]. Fortunately, the algorithm in [RWS11] has a strictly linear complexity and can be easily implemented in parallel. The author therefore believes that the presented importance measure will scale well with the increasing data sizes due to current trend of multi-core CPUs development. The running time of the homological persistence computation could be improved by implementing the algorithm proposed in [dSMVJ10].

5.4 Summary

This chapter presented a novel importance measure for minima and maxima of 2D scalar fields called scale space persistence. This method is based on the mathematical concepts scale space, homological persistence, discrete Morse theory, and combinatorial feature flow fields. A brief introduction to these topics which stem from different areas of mathematics was provided. The combination of these powerful approaches to data analysis resulted in an importance measure for extremal points with many useful properties. As was shown in Section 5.3, the method is the first persistence based importance measure that can deal with noisy data containing outliers. Note that this importance measure assigns a value to all extremal points that exist in the *original* data – one does not need to apply any kind of preprocessing to achieve the results shown in Figures 5.3.3 and 5.3.4.

Due to the concept of the pruned merge graph one does not need to apply any persistence simplification to the combinatorial gradient fields to reliably track the extremal points through the scale space. This is a significant advantage compared to a straightforward application of the integrated persistence measure proposed in [RKWH11] to the scale space data since it reduces implementational efforts and the specification a simplification threshold is not needed. Since the algorithm is readily implemented in a streaming fashion, it is able to treat large data sets in a timely manner and can take advantage of multi-core CPUs and clusters.

The presented method shows great promise as a first step towards an algorithmic framework for automatic crater detection on large elevation maps. This is an important application since one can infer many properties of celestial bodies using this information. The amount and resolution of such data prohibits manual extraction of the craters. Further steps towards this goal could consist of the use of model specific knowledge. One could also investigate the application of non-linear scale spaces which can be easily integrated into scale space persistence measure. Another possible approach for incorporating the structural information contained in the scale space with the homological persistence concept could be built upon multidimensional persistence theory [CZ09].

Chapter 6

Visualization of Three-Dimensional Morse-Smale Complexes

This chapter presents a model of the MS complex as a general graph structure and describes a query language designed to enable generic feature extraction. It shows how the 2-manifold and 3-manifold components of the MS complex can be used to create compelling images. The chapter includes a presentation of an algorithm and efficient data structures to compute these geometries in a simplification hierarchy, in a manner that allows interactive random access to different levels in the hierarchy. These elements are implemented in a hybrid visualization system combining traditional techniques and direct feature visualization. The presented system gives a possibility to a domain expert to interactively specify and explore the MS complex feature space. As mentioned in earlier chapters, MS complex has become a popular structure for topology description and feature extraction. However, the visualization and direct manipulation on this structure have not been presented before, the user was given only limited possibility to control the exploration process. Through the flexibility of the system which will be described here the user can control and manipulate the underlying topological structure of the data, which makes it easier to achieve, for example, such a domain-specific purpose as feature extraction.

6.1 Query language for the topology description

Query-driven visualization techniques provide a mechanism for extracting relevant features, defined by the query, from datasets. Such queries can be range queries on scalar or vector fields in the data or can be queries against visualization constructs such as isosurfaces. McCormick et al. [MIA+07] described SCOUT which allows a user to define range queries against their data using a data parallel language to form the queries. Stockinger et al. [SSWB05] introduced DEX which uses bitmap indexing to efficiently answer multivariate, multidimensional data queries to provide input to a visualization pipeline.

To allow the interactive topology exploration a query language for extracting an information out of the Morse-Smale complex is introduced in this section. The MS complex is a representation of the function in terms of its flow properties, and the full complex is the basis for a large feature space; the problem of extracting features is reduced to querying this structure. In the presented model, the result of querying the MS complex is a set of nodes and a set of arcs. Since higher dimensional manifolds are each associated with one node, extracting them is reduced to the problem of selecting the nodes that generate them. For example, to extract surfaces separating user-selected basins, a query would be designed to select the 1-saddles separating minima, then ascending 2-manifold geometry would be computed to recover the surfaces.

In this model, interactive exploration of features uses the following pipeline: a base MS complex and hierarchy are computed, and then queries on this structure are evaluated by consumer objects. A consumer object can be a renderer, a histogram generator, or any other visualization or analysis tool. The elements of the query language describe the components of the visualization pipeline, which can be exchanged or modified in real time, allowing for the interactive exploration.

6.1.1 Query Object and Function Types

Table 6.1 specifies objects and query functions for generic feature extraction. The *objects* describe a generic data structure, in which an MS complex is represented as a set of nodes and a set of arcs. Each node contains a list of arc references that have it as an endpoint, and each arc references the nodes at its upper and lower endpoints. Both arcs and nodes have

attribute objects, collections of scalars. In the following examples, some of the attributes queried for nodes are index of criticality, function value, and coordinates, although this system could easily be extended to store ascending/descending manifold surface area or volume, count of incident arcs, proximity of other critical points, or other user-specified measures. At the most basic level, an MS complex is represented as an undirected graph where each node and arc has a set of associated values.

A query starts with a precomputed MS complex object. A hierarchy object can be computed by repeated cancellation of critical point pairs of an MS complex. The homological persistence value, mentioned earlier in Section 2.1.4, is used for the topology simplification here. A *persistence selector* operates on a hierarchy and returns the MS complex simplified to an input scalar value. The data structures necessary to extract geometry from the simplified complex are covered in Section 6.2.3. The set of nodes and arcs for a particular MS complex are returned by *node extractor* and *arc extractor* objects, respectively.

Function objects operate on elements of an MS complex. These objects are used to "navigate" on the complex as well as filter desired features. Features are often defined as objects that satisfy certain conditions, and the *node selector* and *arc selector* function objects enable this functionality. These filter functions take as input a *test*, a Boolean function object that decides whether or not a node or arc is to be included in the output of the selector. The test performs a comparison with the value returned by a *value extractor* object applied to a node or arc. The value extractor returns one of the attributes of the node or arc. For example, to extract the minima of a particular MS complex, a node selector is applied to the node extractor, with a test function comparing the index of the node (returned by a value extractor) to zero.

Navigation on the MS complex is equally important in selecting features. These functions correspond to a "walk" on the graph described by the nodes and arcs. The *incident arcs* function object applied to a set of nodes returns the set of arcs touching each node in the set; similarly, the *incident nodes* returns the set of nodes that are endpoints of the input set of arcs. Special variants of these, for example, an *upper incident arcs* and *upper incident nodes* function, denoted $i_a^+()$ and $i_n^+()$ respectively, combine an index test with the incidence operator to return the set in the "upwards" direction. One can then define an *ascending boundary selector* object to find all the critical points on the boundary of the ascending manifold of a critical point by repeatedly applying the upper incident nodes and

Objects		
object	*type*	*name*: *description*
$v ::= (x_0, x_1, \ldots, x_k)$	V	*attribute*: a tuple of values, or functions that evaluate to a value *e.g.*, function value, index, persistence, statistics, arc count, etc.
$n ::= (v, *\ell_a)$	N	node: a pair with an attribute and a reference to a set of incident arcs
$a ::= (v, *n, *n)$	A	arc: a three-tuple of an attribute and two node references
$\ell_n ::= \{*n_0, *n_1, \ldots, *n_{m-1}\}$	L_N	node set: a set of m node references
$\ell_a ::= \{*a_0, *a_1, \ldots, *a_{m-1}\}$	L_A	arc set: a set of m arc references
$msc ::= (*\ell_n, *\ell_a)$	MSC	MS complex: a pair containing an arc set reference and a node set reference
$h ::= (*msc_0, *msc_1, \ldots, *msc_n)$	H	hierarchy: a sequence of MS complexes where msc_{i+1} is obtained by canceling an arc in msc_i

Function Objects		
type	*pseudocode*	*description*
$P : \mathbb{R} \times H \to MSC$	$p(v,h) = h.cancel\_to(v)$	*persistence selector*: returns the MS complex simplified to persistence v
$E_N : MSC \to L_N$	$e_N(m) = m.\ell_n$	*node extractor*: returns the set of nodes in an MS complex
$E_A : MSC \to L_A$	$e_A(m) = m.\ell_a$	*arc extractor*: returns the set of arcs in an MS complex
$F : v \to \mathbb{R}$	$f(v,i) = v.x_i$	*value extractor*: returns a value within an attribute
$T : F \times \mathbb{R} \to \{true, false\}$	$t(f,c) = compare(x,c)$	*test*: compares the result of a value extractor to a constant
$S_a : L_a \times T \to L_a$	$s_a(\ell_a, t) = gather(map(t, \ell_a))$	an *arc selector* returns the subset of its input arcs that pass the test
$S_n : L_n \times T \to L_n$	$s_n(\ell_n, t) = gather(map(t, \ell_n))$	a *node selector* returns the subset of its input nodes that pass the test
$I_n : L_a \to L_n$	$i_n(\ell_a) = \bigcup_{*a_j \in \ell_a} \{a_j.n_0, a_j.n_1\}$	an *incident nodes selector* returns the set of nodes incident to the input arcs
$I_a : L_n \to L_a$	$i_a(\ell_n) = \bigcup_{*n_j \in \ell_n} n_j.\ell_a$	an *incident arcs selector* returns the set of arcs incident to the input nodes

Table 6.1: A functional description of objects and selectors for querying an MS complex. The objects describe generic data structures, and the selector functions extract sets of arcs and nodes.

arcs function. Therefore, the ascending boundary selector computes the critical points on the boundary of a basin of a minimum by applying the upper incident nodes selector to the upper incident arcs selector repeatedly, to return the set of 1-saddles connected to the minimum, the set of 2-saddles connected to those, and then the set of maxima connected to the 2-saddles.

6.1.2 Consumers

A consumer object evaluates a query and converts the resulting list of arcs or nodes into another format. Different kinds of consumers are: geometry processors, renderers, and statistics generators. A consumer may rely on the input gradient field and any computed attributes to do its work. For example, a geometry extractor, described in Section 6.2, may take as input a query that returns a set of saddles, and outputs the set of ascending and descending 2-manifold surfaces. A render object typically is at the end of the visualization pipeline, and can display a set of nodes, a set of arcs, a set of geometry extractors, the scalar function value volume, and an index volume, and render the scene as described in Section 6.3.3. A statistics generator could output a histogram of an attribute value for a set of arcs and nodes. The consumer objects are not listed in Table 6.1, since they are highly application dependent. For example, in the current implementation, the interest lies in generic visualization of extracted features, and therefore geometry extractors and a renderer were implemented. However, one can envision a usage scenario where an application scientist wants to perform custom statistics on a set of features extracted at multiple time steps of a simulation. In this case, the implementation of a statistics tool consumer is left up to the user.

6.2 Geometry computation

To be able to represent the topological elements: nodes, arcs, descending/ascending manifolds visually, their analytical description has to be converted into the geometric one. Computing ascending and descending manifolds in the context of discrete Morse theory has been well studied: first Cazals et al. [CCL03] computed them from a tree-based representation of the gradient, then Gyulassy et al. [GBHP08] gave a description of how to compute them by searching in the discrete gradient field. Neither of these techniques take into account the need for random access into a hierarchy of MS complexes. In previous implementations [CCL03], simplification is achieved by repeated reversal of gradient paths, and therefore random access between levels of a hierarchy would require sequential reversal of paths and expensive re-computation of the manifolds in the modified gradient. The data structures and algorithms presented in this section do not require modification of

the input gradient field, and avoid re-computation of manifolds during interactive exploration. First, algorithms to compute ascending and descending manifolds are reviewed. Next, it is shown how a manifold is translated to renderable primitives. Finally, a data structure is presented together with a technique for maintaining manifold geometry efficiently in a simplification hierarchy.

The nodes, arcs, and higher dimensional manifolds extracted from the discrete gradient field are initially represented as sets of cells and are transformed into geometric primitives for rendering. For example, spheres of different colors are used as placeholders at the barycenters of critical cells, since they are a well-understood metaphor for critical points. Tubes and semitransparent surfaces are used for 1- and 2-manifolds respectively.

6.2.1 Computing Ascending/Descending Manifolds

This section re-states the algorithm described by Gyulassy et al. [GBHP08] for computing ascending and descending manifolds. The following algorithm collects the d-cells in the descending manifold of an index-d critical cell β by repeated application of the flow operator defined in Section 2.1.3. The following pseudo-code implements this operator, isTail() returning true if a cell is the tail of a gradient arrow and false otherwise, isHead() returning true if a cell is the head of a gradient arrow, and discreteTangent() applied to a cell returning the cell it is paired with in the discrete gradient.

```
1: AddDescendingCells(cell β) :
2: result = { β }
3: for α ∈ ∂β do
4:    if isTail(α) then
5:       β_next = discreteTangent(α)
6:       result = result ⋃ AddDescendingCells(β_next)
7:    end if
8: end for
9: return result
```

The input to the algorithm is the critical cell, and the output is the set of cells of the same dimension that form its descending manifold. The algorithm to compute ascending manifolds is similar, replacing the boundary operator with its inverse.

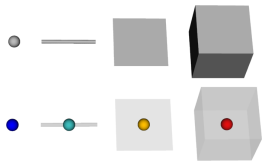
```
1: AddAscendingCells(cell α) :
2: result = { α }
3: for β ∈ ∂^{-1}α do
4:    if isHead(β) then
5:       α_next = discreteTangent(β)
6:       result = result ⋃ AddAscendingCells(α_next)
7:    end if
8: end for
9: return result
```

6.2.2 Renderable Geometry from Manifolds

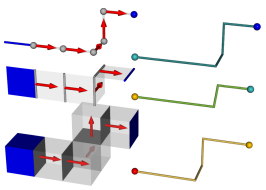
The nodes, arcs, and higher dimensional manifolds extracted from the discrete gradient field are initially represented as sets of cells, and are transformed into geometric primitives for rendering. For example, spheres of different colors are used as placeholders at the barycenters of critical cells, since they are a well-understood metaphor for critical points. Figure 6.2.1 shows the geometry conversion for nodes, arcs, ascending and descending 2-manifolds, and ascending and descending 3-manifolds.

6.2.3 Maintaining Geometry in a Hierarchy

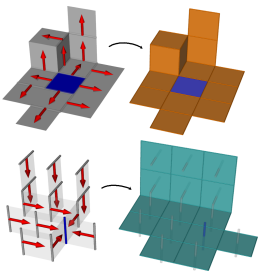
A simplification hierarchy records a sequence of cancellations of pairs of critical points. The geometry associated with the ascending and descending manifolds of any critical points neighboring a cancellation will change. Forman [For98] showed that a cancellation can be realized in a discrete gradient field by simply reversing the discrete vectors on a V-path. In this setting, the ascending and descending manifolds can be recovered by the algorithm presented in Section 6.2.1. However, in a practical system where interactive browsing of the hierarchy is desired, such an operation is very costly. Furthermore, anti-cancellations become just as expensive. Gyulassy et al. [GNP$^+$06] introduced data structures for maintaining the geometry of arcs in a cancellation sequence, however, this structure did not handle the anti-cancellations necessary to browse a hierarchy. The data structures for maintaining manifold geometry introduced in this section follow a similar



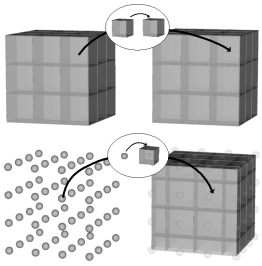
Critical cells are displayed with spheres centered at their barycenter. A blue sphere is used to indicate a minimum, 1-saddles are cyan, 2-saddles are yellow, and maxima are red.



Arcs are V-paths in the gradient vector field, with each index-d + 1–index-d arcs displayed as lines with vertices at the barycenter of each d-cell.



A descending 2-manifold (top) is displayed as the surface formed by the quads. An ascending 2-manifold (bottom) is displayed as the surface formed by the dual quads to the 1-cells.



Each 3-cell in the data is uniquely part of one descending 3-manifold, and the 3-cells themselves are the voxels (top). A vertex is unique to a single ascending 3-manifold, and its dual 3-cell becomes a voxel (bottom).

Figure 6.2.1: Once topological structures are computed as sets of cells, they are converted into geometric primitives for rendering.

merging structure, and define a similar acyclic graph. In a common usage scenario, a user may wish to skip over tens of thousands of cancellations to view the hierarchy at different persistences. Using the presented approach, a process that previously took several minutes becomes interactive.

This section will proceed with a presentation of an efficient technique for representing the geometry of ascending and descending manifolds at any time in the hierarchy by storing their merging in an acyclic graph. Gyulassy el al. [GBHP08], characterized how a cancellation changes both the structure of the complex and the geometry of the manifolds. When n_u, an index-$i+1$ critical point, and n_l, an index-i critical point, are canceled, the following changes occur to the manifolds:

(1) For every node of index $i+1$ in the neighborhood of n_l, merge its descending manifold with the descending manifold of n_u.

(2) For every node of index i in the neighborhood of n_u, merge its ascending manifold with the ascending manifold of n_l.

Using this foundation, the data structure creates a "merge" element every time the manifold of a node changes. A hierarchy is a simplification sequence, therefore one can assign an integer time to each creation/destruction event. In particular, the creation time of a merge element is equal to the number of cancellations so far in the sequence. The merge elements are defined as follows:

$$merge ::= (base\_merge,\ other\_merge,\ time) \mid leaf\_geometry$$

A merge object is either a 3-tuple of two merge objects and a merge time, or a pointer to the leaf geometry. Leaf geometry associated with a critical cell α is evaluated as the result of AddDescendingCells(α) or AddAscendingCells(α), and has an implicit merge time of 0. Every node n in the MS complex has one merge object for its ascending manifold geometry, denoted $n.asc\_man\_geom$, and one for its descending manifold geometry, $n.dsc\_man\_geom$. Initially, before any cancellation has occurred, every merge object is leaf geometry. During a cancellation new merge nodes are created and the graph structure is updated. In the following algorithm, neighborhood() of a node returns the set of nodes that share an arc in the complex.

```
1: ManifoldCancellationUpdate(node n_l, node n_u, int cancel_time) :
2: for n_i ∈ neighborhood(n_l), n_i ! = n_u do
3:     new_dsc_geom = (n_i.dsc_man_geom, n_u.dsc_man_geom, cancel_time)
4:     n_i.dsc_man_geom = new_dsc_geom
5: end for
6: for n_i ∈ neighborhood(n_u), n_i ! = n_l do
7:     new_asc_geom = (n_i.asc_man_geom, n_l.asc_man_geom, cancel_time)
8:     n_i.asc_man_geom = new_asc_geom
9: end for
```

The history of the changes to a node's geometry is stored in the *base_merge* field of the merge element: it is a list with the creation times of each merge object. Therefore, to extract the state of the merge graph at time t, one has to start at a node's merge element pointer, the newest merge node, and follow *base_merge* pointers until *cancel_time* is at

most t. The merge element returned was the merge element pointed to by the node at time t in the creation of the hierarchy. Since directed edges in the merge graph always point from higher cancellation-time merge objects to lower cancellation-time objects, any path in this graph is monotonic, and hence the graph is acyclic. To recover the full ascending or descending manifold from a merge element, one has to gather the leaf geometries that are alive at the current simplification level with a depth-first search. The following algorithm returns the set of merge leaves that have been merged in the cancellation process to create the input merge element. In this algorithm, isLeaf() returns true if a merge element is a pointer to leaf geometry, false otherwise.

```
1: RecGatherOddLeaves(merge m, set& s) :
2: if isLeaf(m) then
3:    if s ∩ {m} ! = {} then
4:       s = s − {m}
5:       return
6:    else
7:       s = s ∪ {m}
8:       return
9:    end if
10: end if
11: RecGatherOddLeaves(m.base_merge, s)
12: RecGatherOddLeaves(m.other_merge, s)
```

Note that only leaves that have been visited an odd number of times have to be counted: this simulates symbolically reversing the path along an arc during cancellation. Figure 6.2.2 illustrates this: a single cancellation reverses the flow along an arc and extends the manifold for neighboring nodes; a second cancellation reverts those cells to their original direction. The following algorithm shows how to traverse a hierarchy to extract descending manifolds for a particular time in the simplification sequence.

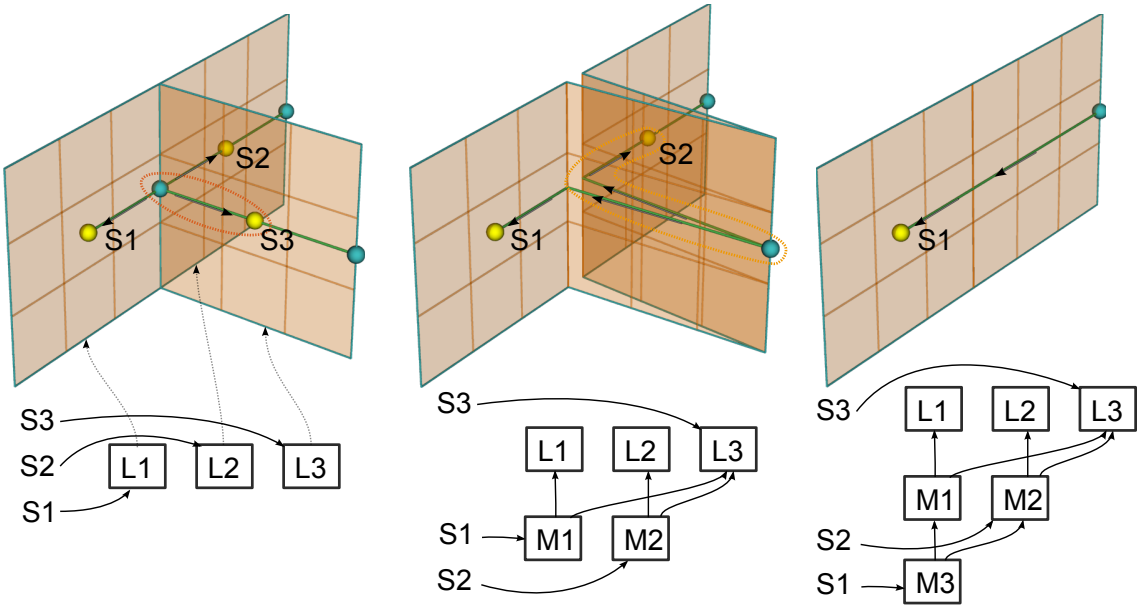


Figure 6.2.2: S1, S2, and S3 are 2-saddles, with initial leaf geometry merge elements L1, L2, and L3, respectively (left). The first cancellation of S3 and the circled 1-saddle creates new merge elements M1 and M2, both having $time = 1$, for the nodes S1 and S2 (middle). The second cancellation, S2 with the circled 1-saddle, creates a new merge element M3 for S1 with $time = 2$ (right). The surface of S1 at $time = 1$ is L1 and L3. At $time = 2$, the surface from S1 is L1 and L2; L3 is counted an even number of times in a depth-first search.

1: DescendingManifoldAtTime(node n, int $time$) :
2: $m = n.dsc\_man\_geom$
3: **while** $m.time > time$ **do**
4: $\quad m = m.base\_merge$
5: **end while**
6: $s = \{\}$
7: RecGatherOddLeaves(m, s)
8: return s

Figure 6.2.2 illustrates these algorithms and data structures, and shows how they are maintained through some cancellation operations.

6.3 Rendering of the topology

In this section an interactive feature extraction and visualization systems are described as a practical example. These consist of two main components: an interface for constructing queries, modifying parameters and assigning rendering attributes, and a renderer that is specially designed to handle the kinds of features that can be identified using the MS complex.

6.3.1 Interactive Exploration

The data objects of an MS complex and function objects in a query have a natural branching and dependency structure, and the presented user interface utilizes this by representing a scene as a work flow diagram. This prototype implementation supports specifying queries in a graphical user interface, where the input and output to selectors are represented by arrows. The user chooses different types of objects (tests, selectors, etc.) from a tool bar and connects them together in a work flow graph. Each object and selector function has properties that can be manipulated. For example, the constant factors in a test function that implements range comparison can be changed with sliders to interactively adjust queries, and any changes force recomputation of downstream queries. The number of objects and depth of the work flow diagram are limited in practice, and each downstream object can be recomputed interactively. The interface provides for saving and loading an interaction session in an XML file. This allows a scientist to use the GUI to design queries, and then use the resulting XML description to extract features in a batch job, for example, to dump surfaces or statistics to disk for several time steps of a simulation. Figure 6.3.1 shows an interaction session in the interface prototype.

6.3.2 Rendering Attributes

A work flow diagram is also the scene graph and gives the user control of rendering attributes. Each query object of the MS complex can be assigned valuator functions that determine how its result is rendered by a render object. For example, a set of nodes can be assigned a valuator object that scales the radius of its rendered sphere by some transfer function. As another example, the surface output from a geometry extractor can

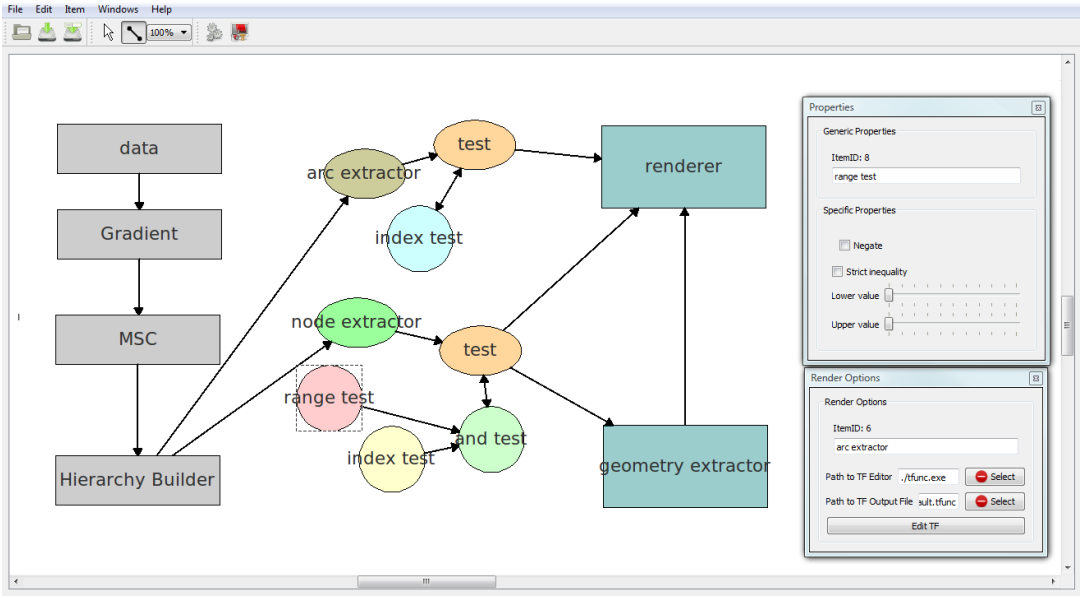


Figure 6.3.1: A user specified work flow diagram is translated into queries on the MS complex. Updates to the diagram are reflected in real time in the output of the renderer. Here a simple scene is illustrated. A gradient, MS complex, and hierarchy are pre-computed from the input data. The result of node and arc extractors are filtered by selectors. Each selector is also linked with a Boolean test. The range test is selected, which brings up a window to manipulate its properties.

be rendered with its own transfer function. In this way, one can assign valuators to sets of nodes, arcs, surface, or volumes that are locally evaluated by each element in that set. Initially, the nodes and arcs extracted from an MS complex have default values for rendering attributes, such as coloring nodes by index. In the presented model, a new selector or node inherits its valuators from its parent.

6.3.3 Interactive Rendering

The developed rendering system combines traditional volume rendering with direct feature visualization. A depth-peeling GPU ray-caster [BPaLDCS06] allows inlays of semi-transparent solid geometry (spheres, lines, tubes, surfaces) into a locally-controlled volume rendering. The overriding design principle to this rendering system is to enable rendering of any feature using its own rendering attributes combined with the underlying scalar field. To achieve this, the implementation uses (1) the input volume of scalar values, (2) a volume of ascending or descending 3-manifold identifiers storing the segmentation

of the domain, (3) a set of surfaces representing ascending and descending 2-manifolds, (4) a set of lines or tubes representing arcs, and (5) a set of spheres representing nodes. The following describes how each of these is handled.

Volume Rendering: In the developed volume ray caster, the four-channel colors (red, green, blue, and alpha) of sample points are found along a ray and composited front-to-back. The presented system allows the user to select the transfer function, color, and blending mode for each topological feature individually. Furthermore, a user can rescale the function values within ascending or descending 3-manifolds, for example, to display ones with different ranges uniformly with the same transfer function.

In the following pseudocode that returns the color of a sample in the volume, let $f : \mathbb{R}^3 \to \mathbb{R}$ be a function that returns the value of the input scalar field at the point, let $i : \mathbb{R}^3 \to \mathbb{Z}$ be a function that maps the point to the unique ID of the ascending or descending 3-manifold containing it, let $rmin : \mathbb{Z} \to \mathbb{R}$ and $rmax : \mathbb{Z} \to \mathbb{R}$ map an ID to local rescale values, let $color : \mathbb{Z} \to \mathbb{R}^3$ map an ID to a constant color, let $transfer : \mathbb{Z} \to (\mathbb{R} \to \mathbb{R}^3)$ map an ID to a transfer function, and finally let $blend : \mathbb{Z} \to [0,1]$ map an ID to a constant controlling the compositing of $color$ with $transfer$.

RGBA(point p):

1: $ID = i(p)$
2: $VAL = (f(p) - rmin(ID))/(rmax(ID) - rmin(ID))$
3: $RES.rgb = blend(ID) * transfer(ID)(VAL).rgb +$
$(1 - blend(ID)) * color(ID)$
4: $RES.a = transfer(ID)(VAL).a$
5: return RES

The result of $f(p)$ is rescaled by the feature rescale values found at ID $i(p)$ of the *rmax* and *rmin* tables, and this value is sent to the transfer function found for ID, returning a color. This color is blended with the assigned color of the feature, using the blend factor and color found for the ID in the $blend$ and $color$ tables respectively.

Surface Rendering: Surface colors are computed similarly to colors in a volume, except that every point on a surface has the same ID, and therefore the functions *i*, *rmin*, *rmax*, $color$, and $blend$ are all replaced by constants, and $transfer$ returns the same transfer function for all points on the surface.

Tubes and Lines: Arcs are rendered as tubes or lines. A tube or line is represented as a list of vertices, each having a color and radius. Tubes and lines are fully opaque objects. The values for the color and radius along a tube or line can be set using a transfer function.

Spheres: Nodes are rendered as spheres. A sphere may have arbitrary color and radius. Spheres are fully opaque objects.

The rendering system utilizes a GPU-based ray caster and depth peeling to resolve transparency. Samples in the volume and on surfaces and lines are shaded with fragment programs written in GLSL.

The functions f and i are three-dimensional samplers operating on volumetric textures storing the scalar values and indices, respectively. The index volume i is sampled with no interpolation. A single transfer function is implemented as a one-dimensional texture lookup, and $transfer$ is implemented as a two-dimensional lookup. An additional mapping $texID : \mathbb{Z} \to \mathbb{Z}$ is used to map volume indices to transfer function ID, and the result of this lookup is the first coordinate in the two-dimensional transfer function texture. The other coordinate is simply the rescaled function value. Note that the maximum index number is the number of extrema generating the segmentation of the volume. GLSL restricts the maximum size of a texture dimension to 4096, therefore $texID$, $rmin$, $rmax$, $color$, and $blend$ are implemented as linear lookups into two-dimensional textures. Note that one can apply Laplacian smoothing to lines and surfaces for aesthetics.

As the MS complex is simplified, several of the regions in the index texture will merge. Instead of updating the full volumetric texture, the merging is simulated by copying the parameters to the two-dimensional textures. This avoids having to modify the volume value and volume index textures.

6.4 Results/Examples

The visualization results were performed on an off-the-shelf 2.26GHz laptop with 4GB main memory and GeForce GT 130M graphics card with 1GB physical memory. In each case, the 1-skeleton of the MS complex was precomputed using a single-block version of the algorithm presented by Gyulassy et al. [GBHP08], with pre-processing run times of the same order, taking between one second (Tetrahedrane) to four minutes (Cosmology).

Subsequent exploration, in terms of query evaluation and parameter editing, is performed interactively, with frame rates varying between 30 frames per second for smaller, less complex scenes, to .5 seconds per frame for large scenes with high level of transparent overlap. The presented renderer is unoptimized research code, and one can expect that frame rates can be improved.

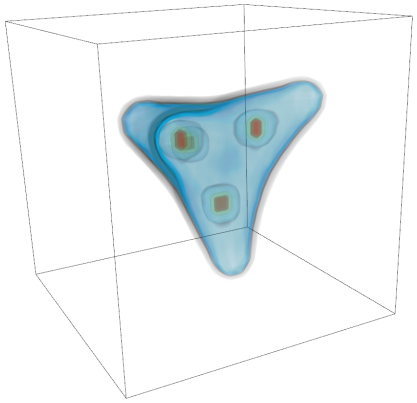
The overall memory requirements of this approach are output sensitive, however, every analysis pipeline begins with computation and storage of the discrete gradient field, taking exactly 8 bytes per sample location. Furthermore, the entire dataset is loaded into main memory and GPU texture memory. If segmentation volumes are needed in the pipeline, they are also computed and stored in texture memory, taking 4 bytes per sample location. Nodes, arcs, and geometry objects are computed on-the-fly; for example, the *impact crater* surface of the porous solid required storing 30K triangles and function values, replicated on the GPU.

The examples in Figures 6.4.1 – 6.4.6 show scenes highlighting the illustrative power and generality of the presented technique in order of increasingly complex topology-rich visualizations. The *Tetrahedrane* (Figure 6.4.1) has simple topology and is used to illustrate two rendering modes, showing the structure and the geometry of the MS complex. The *Silicium* example (Figure 6.4.2) illustrates how simple queries can be composed to construct an abstract representation of the molecule. In the *Combustion simulation* (Figure 6.4.3), the extraction of 3-manifold basins around minima hypothesized to be involved in flame extinction events is performed, and one can see how persistence simplification allows exploration of these features at multiple topological scales. With the *High-pressure H_2O model* (Figure 6.4.4), one can see how the MS complex provides a more accurate interpretation of features: how topological pouches can be extracted, something not possible with previous approaches. In the *Porous medium* dataset (Figure 6.4.5) it is illustrated how to build non-physical features, such as the crater surface derived from the impact with a micro-meteoroid, or the most likely shear fracture dividing the top and bottom of the material. The last example (Figure 6.4.6) shows how the same analysis of a *Cosmology simulation* can be visualized in drastically different ways: in one, the maxima of density are rendered as spheres whose size and color depend on function value; in the other the largest simple contours around a maximum are displayed with color according to function value. Overall, this set of examples illustrates the power of the presented topology-rich

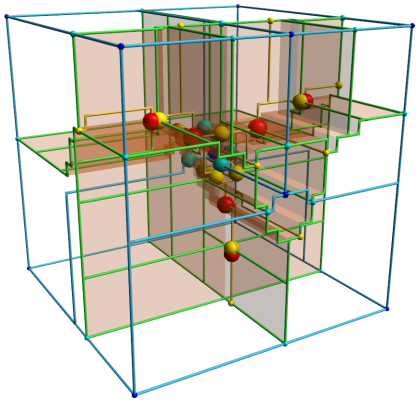
visualization technique, showing a variety of visualizations that were not available within a single integrated system.

6.5 Summary

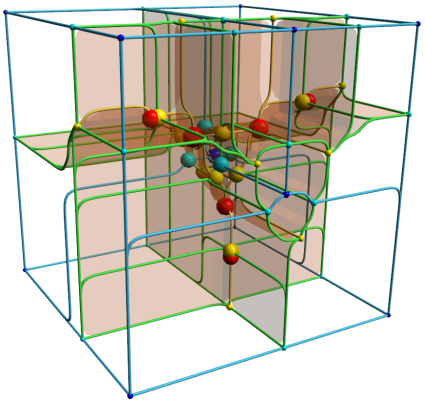
Topological methods for feature extraction are considered to be heavy and hard to understand. It makes topology extraction a very domain-specific task. This chapter presented a tool for a general interactive topology analysis and exploration. It was shown that the domain expert can extract more knowledge from data if given a flexible possibility to control the visualized structures. A generic query language to acquire different structures of an MS Complex at different levels of hierarchy was developed. The introduced data structures help to maintain the simplification hierarchy and interactively access its parts for the visualization. A scene graph based rendering system helps to concurrently visualize the features of different dimensions, controlled by various attributes. Although some computation algorithms already exist in the literature, the visualization of the features shown in this chapter was never developed before. The system presented here is the first attempt to visualize topology using three-dimensional Morse-Smale complex structure. The extension of it to allow for the exploration of other topological structures is pretty straightforward. To achieve different goals, the new functional or visualization components can be combined with the existing ones.



(a) Tetrahedrane dataset.

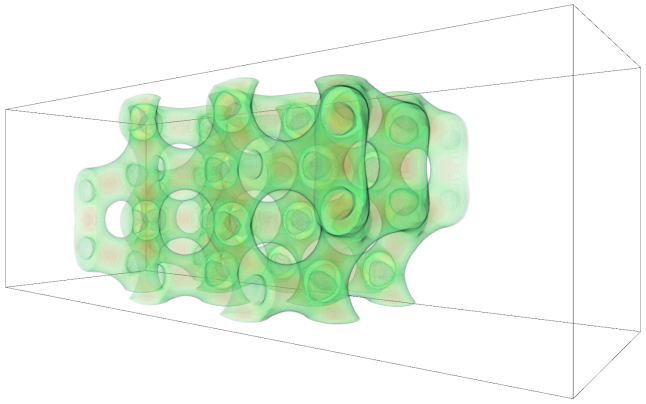


(b) Structure: Add all nodes and arcs from the complex to be rendered. Set radius of the nodes to be a scale factor of the function value. Select all 1-saddles and extract their ascending 2-manifold geometry, render orange with transparency.

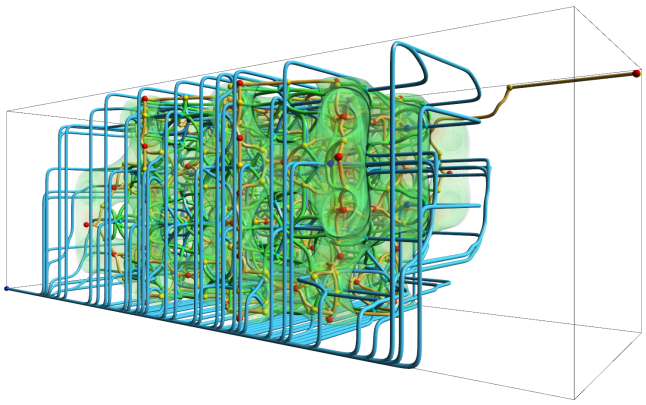


(c) Geometry: Add all nodes and arcs from the complex to be rendered. Set radius of the nodes to be a scale factor of the function value. Select all 1-saddles and extract their ascending 2-manifold geometry, render orange with transparency. Lines and surfaces with three iterations of smoothing.

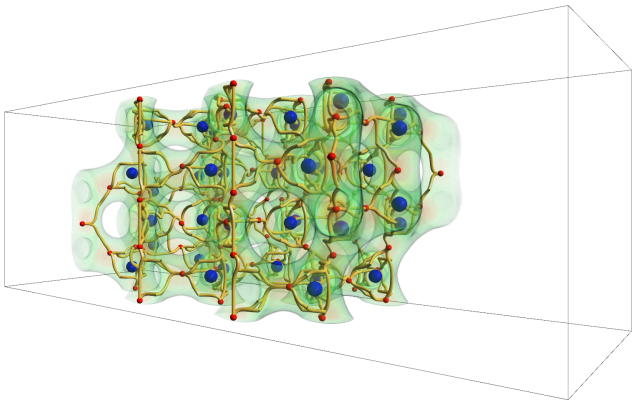
Figure 6.4.1: Tetrahedrane dataset, $24 \times 24 \times 24$ floats: The scalar value represents the probability distribution electrons in a tetrahedrane (C_4H_4) molecule. High peaks are centered around the atoms. Selected elements from the computed MS complex are shown, first without geometric smoothing (b), and after some iterations of Laplacian smoothing (c).



(a) Silicium dataset.

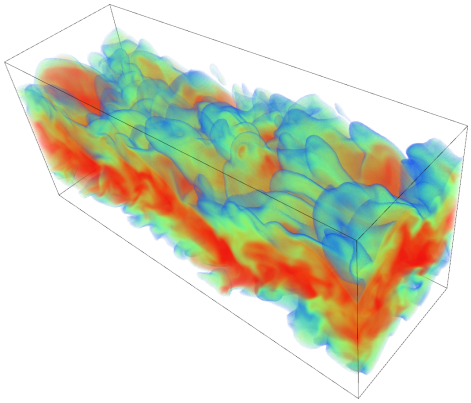


(b) Full MS complex: Create a hierarchy simplifying to 10% total persistence. Select all arcs and nodes from the hierarchy and render them.

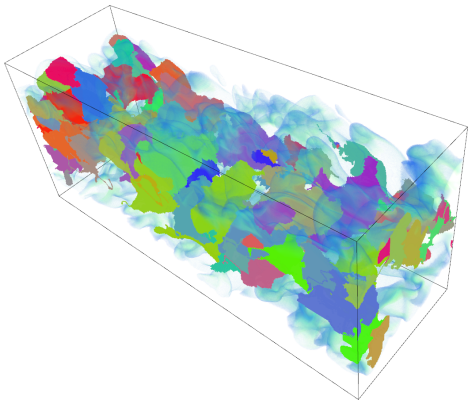


(c) Peaks and ridges: Create a hierarchy simplifying to 10% total persistence. Select all 2-saddle-maximum arcs and add them to be rendered with smoothing. Select maxima incident to these arcs and minima between 15.0 and 35.0, add to rendering. Atom locations (blue) and the tetrahedral arrangement of covalent bonds (red) around them (right) are shown.

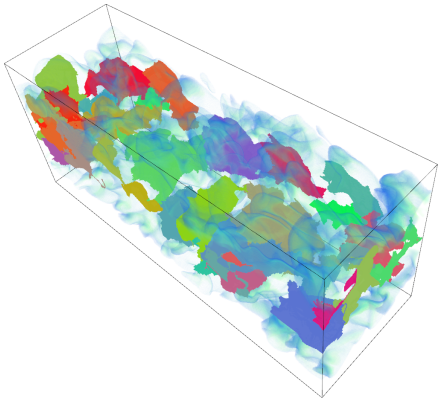
Figure 6.4.2: Silicium dataset, $34 \times 34 \times 98$, byte: In the simulation of a silicon lattice, values correspond to electric potential.



(a) Combustion jet dataset.

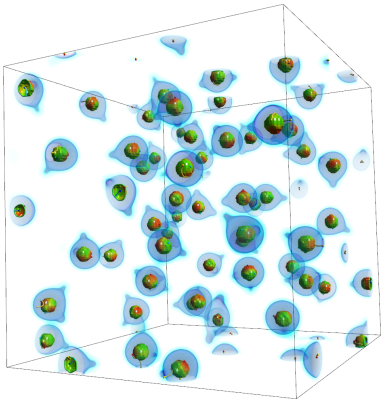


(b) Persistence 2.5%: Create a hierarchy simplifying to 2.5% total persistence. Select all minima in the value range $[0.4, 0.6]$, and assign a random color. For each minimum, compute its ascending 3-manifold. Render each selected 3-manifold as fully opaque, with the associated minimum's color. Unselected 3-manifolds are rendered with a background transfer function.

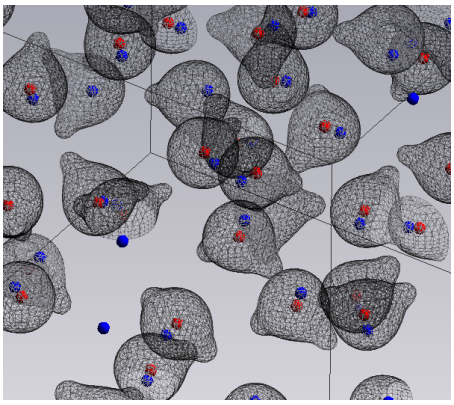


(c) Persistence 5%: Create a hierarchy simplifying to 5.0% total persistence. The selection and rendering is subsequently the same as (b). The background transfer function for both is a ramp from blue (low value) to green to red (high values)

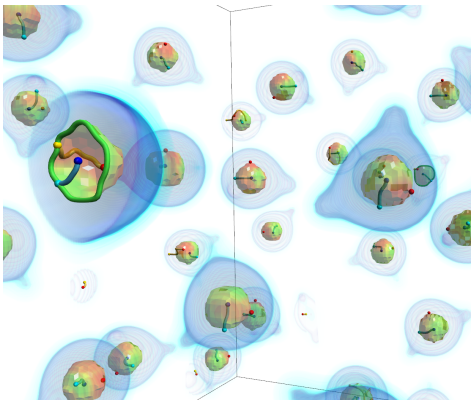
Figure 6.4.3: Combustion Jet, $384 \times 168 \times 124$, float: In the simulation of a combustion jet, fuel mixture fraction basins have been linked to dissipation elements. These volumetric elements are rendered at two topological scales. The basins at the finer scale (b) merge into larger basins at the coarser scale (c). High values in the transfer function are red, low are blue.



(a) High-Pressure H_2O dataset.

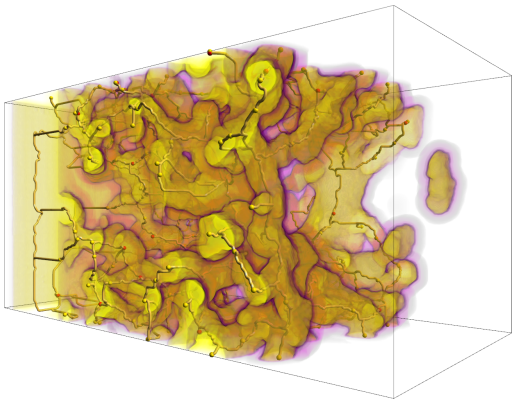


(b) Contour tree based features: Applying contour-tree based analysis [PCMS09] to the same data, a persistent minimum/maximum pair without the separating surface suggests a dipole, not a pouch.

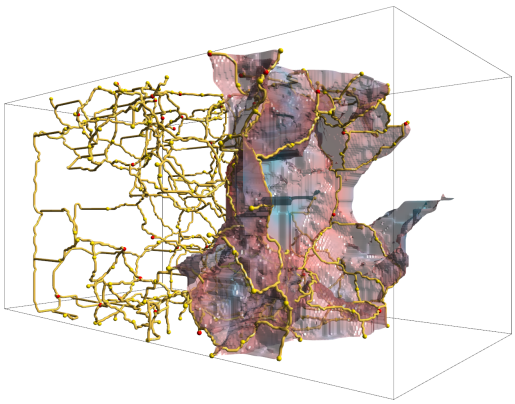


(c) MS complex pouches: Create a hierarchy simplifying to 5.0% total persistence, select all extrema and arcs entirely above 0.2 for rendering. Extract ascending 2-manifolds for any 1-saddles selected, smooth, and apply a green (low) to red (high) transfer function with transparency.

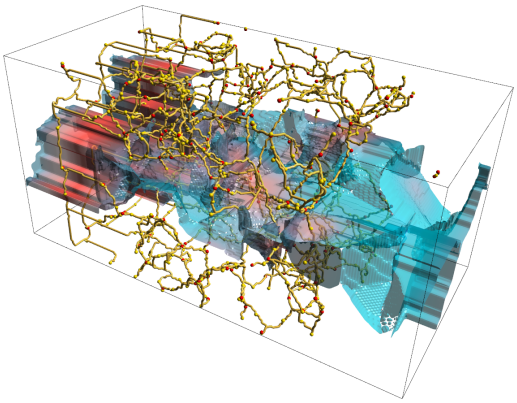
Figure 6.4.4: High-Pressure H_2O dataset, $128 \times 128 \times 128$, float: The scalar value is electron density distribution in simulated high-pressure water. The high values forming a shell around the location of the oxygen atom. This topological pouch is extracted and the surface is colored with a separate transfer function.



(a) Porous Solid dataset.

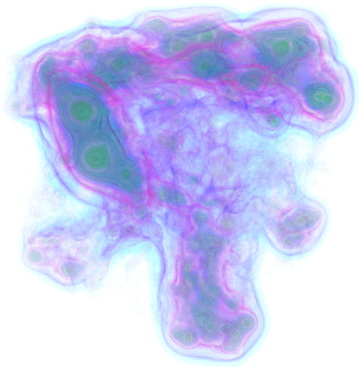


(b) Core Structure and Impact Crater: Simplify to 5.0% total persistence using the threshold values found by Gyulassy et al. [GDN+07]. Select 2-saddle-maximum arcs in the range $[-2.0, 30]$ and incident nodes for rendering. Select the lowest minimum (located outside the crater, and select neighboring 1-saddles (using two incidence selectors). Extract ascending 2-manifolds from these saddles, smooth, and apply a blue (low) to red (high) ramp transfer function with transparency.

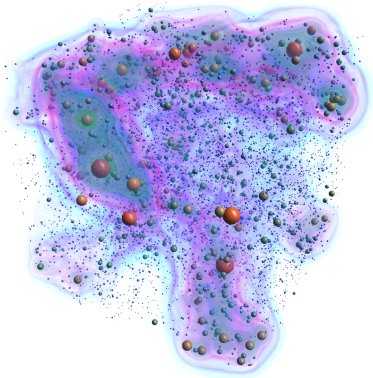


(c) Fracture Surface through the weakest points connecting the top half of the material to the bottom: Use the same hierarchy and arcs/nodes for rendering as in (b). Select all 2-saddle-maximum arcs that cross y-coordinate = 57.5. Select 2-saddles from their incident nodes. Extract descending 2-manifolds from these, smooth, and apply a blue (low) to red (high) transfer function with transparency.

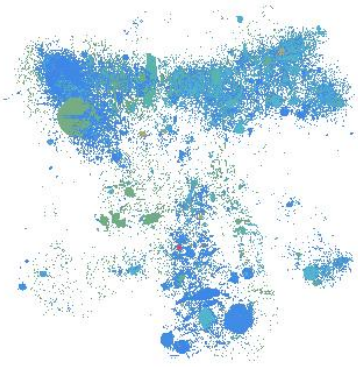
Figure 6.4.5: Porous Solid, $115 \times 115 \times 237$, float: A signed distance field from the material boundary represents a porous copper foam. This time-step ends a sequence where a micro-meteoroid impacts the foam from the right.



(a) Cosmology Simulation dataset



(b) Peaks: Simplify to 5.0% total persistence after a log scaling. Select all maxima in the hierarchy, and add to render. Set the color of the sphere to be based on a transfer function with a blue (low) to red (high) ramp, and the radius also to be scaled by the value. Render the volume with a background transfer function.



(c) Mountain tops: Simplify to 5.0% total persistence after a log scaling. Select all maxima in the hierarchy and extract descending 3-manifolds. Set the local scaling factor on each 3-manifold to: *rmax* = value of maximum, *rmin* = value of highest neighboring 2-saddle. A step function is used to map values to alphas, and the color is set to the color of the maximum.

Figure 6.4.6: , $256 \times 256 \times 256$, float: In this simulation of a 90 MParsec box [HRWH05], the scalar value represents particle density in space. Peaks in this data are displayed in two ways: (b) direct rendering of peaks; (c) locally-scaled transfer functions show the region of influence of a maximum, more correctly displaying the actual volume of a feature.

Chapter 7

Illustrative Visualization: Interrogating Triangulated Surfaces

7.1 Surface interrogation: motivation

One of the major tasks in the process of styling is to design ”visually pleasing” curves and surfaces in a functional or aesthetic way. For example, the roof of a car should be a convex surface without bumps of wiggles. These kinds of features correspond to intrinsic geometric properties of the surface, such as continuity, curvature or torsion. Using conventional rendering algorithms, these surface characteristics are often not visible, because of the smoothing in the interpolative shading algorithms.

In the traditional analytical scenario, reflections are generated by fluorescent light bulbs in long lines along a ceiling. The geometry and the aestetic value of these reflections are used to assess the quality of the manufactured object. With modern graphics hardware and algorithms these reflections can easily be simulated on a computer. Surface interrogation algorithms [Hof90, Hos85, HSG90, Ski89] can be used to visualize surface properties such as convexity, geometric convexity, or curvature to assess the aesthetic quality of a surface.

Basically, there are three classes of interrogation methods:

- Using color maps or textures to visualize scalar values describing quality measures of greatest interest are curvature measures of the surface: mean curvature, Gaussian

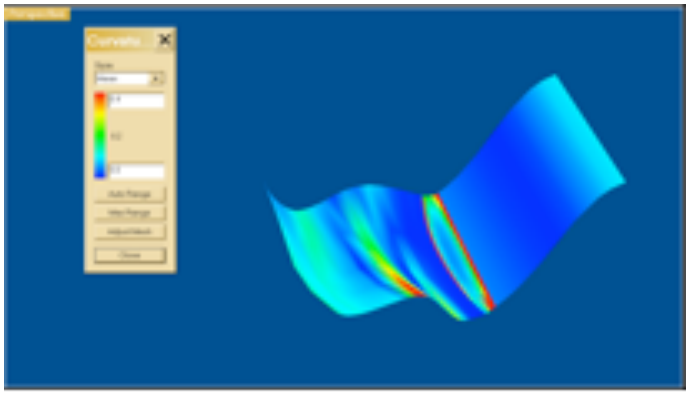


Figure 7.1.1: Mean curvature of a blending surface with tangent continuity vizualized using a spectrum colormap (red indicates areas of high curvature while blue indicates area of low curvature).

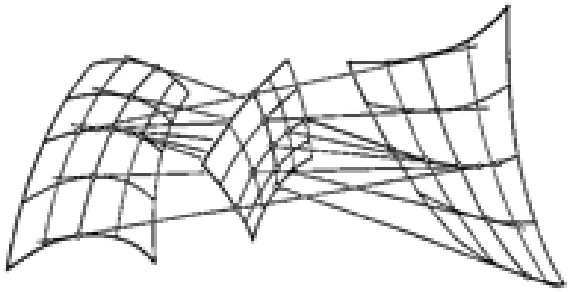
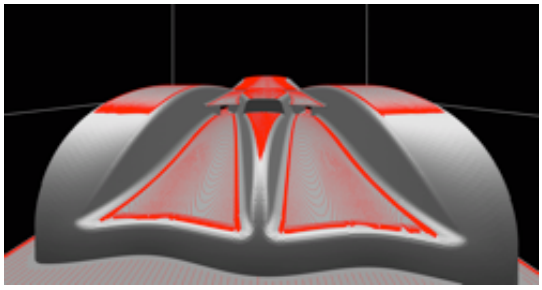


Figure 7.1.2: A surface with its two focal surfaces.

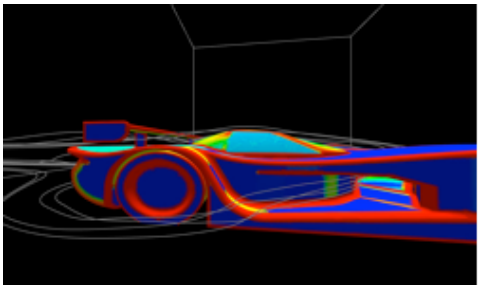
curvature, and min-max curvature. In Figure 7.1.1 the mean curvature of a blending surface with tangent continuity is visualized by a scalar color map;

- Describing the curvature properties of the surface by a computed secondary geometry, such as a generalized focal surface: Figure 7.1.2 shows a surface with its associated focal surfaces. The surface under consideration is in the middle and its two focal surfaces are on either side; and

- Computing curves on a surface, like the reflections generated by long parallel light lines on a real object: in Figure 7.1.3a the reflection lines generated by 25 parallel light lines illuminating the surface of a car model are shown.



(a) The front view with reflection lines generated by 25 parallel lines overhead.



(b) Streamlines from a simulation of the flow around and through a car body with the car surface color-mapped to show surface curvature. Red shows areas of highest curvature, with the areas of least curvature in blue.

Figure 7.1.3: A wind tunnel simulation of a race car.

Surface interrogation techniques are well established and applied on free-form surfaces, such as NURBS surfaces, in both academic and commercial systems. Such systems statically compute characteristics of the free-form surface and visualize them. However, other approaches for combining simulation with visualization are more attractive [CDP+96, CNSD+92, CNSD, HGM02]. In this area for the computation of dynamic characteristics, such as flow along the surface or temperature on the surface, free-form surfaces have to be tesselated into triangular meshes. To allow for concurrent visualization of the surface quality and the flow simulation results, a surface interrogation algorithm has be able to work on the triangulation. This chapter presents such a method and shows results of its practical application.

An example of such a simulation ca be seen in Figure 7.1.3. The figure shows the result of a wind tunnel simulation of a race car. In this simulation, the surface of the car model is tesselated from free-form surfaces into triangle meshes with a total of 400k triangles and the space around the car is divided into millions of tetrahedra. The simulation result is visualized with several streamlines in the vector field; the surface curvature is estimated and visualized with color mapping in the same view volume. Combining the simulation results with the surface interrogation results can give better insight into the surface properties, the flow field, and their intrinsic relationship.

7.2 Curvature estimation over triangulated surfaces

This section presents an approach for computing the curvarture of a triangulated surface. The proposed approach is based on Bézier patches and preserves localized information, which is beneficial for assessing the quality of the tesselation.

7.2.1 Related work

Many surface interrogation methods are based on the curvature estimation of the targeted surfaces. For example, the generalized focal surface is a function of the principal curvatures, k_1 and k_2 of the targeted surface. Therefore, the primary component of these surface interrogation methods is to estimate the curvatures of the triangulated surface.

Curvature estimation on triangulated surfaces has been studied for several years [Ham93, KLM98, Mar98, MDSB02, Pet02, RB05, DW05, AT05, MW00, BCM03, WB01, Tau95]. Krsek et al. [KLM98] and Petitjean [Pet02] have given good introductions to several curvature estimation algorithms and compared some of them: approximation by an analytic surface [Ham93], approximation by curves [Mar98], convolution operators, and discrete curvature methods [MDSB02]. Surazhsky et. al. [MSR07] made a comparison of five methods (Paraboloid fitting [Ham93], Circular Fitting [Mar98], Gauss-Bonnet Scheme (Angular Defect) [MW00], Watanabe and Belyaev Approach [WB01] and Taubin Approach [Tau95]) for estimation of Gaussian and mean curvatures. The simplest method is called the *circular fitting*. It requieres a minimum of 4 points or, more practically, 7 points to form a neighborhood for estimation. More complicated methods usually require more neighboring points to smooth noisy data.

Most of these algorithms are developed to solve the noise problem in measured data, such as range data, so the more neighbor points these algorithms take into account to estimate curvature, the better they smooth the noise and provide good estimates. For surface interrogation, the question is completely different. There is no noise, but irregularity must be considered. The curvature estimation algorithm for surface interrogation should consider as few neighbor points as possible to keep the local surface information as localized as possible.

7.2.2 Curvature estimation method using Bézier patches

In this section a novel curvature estimation algorithm for triangulated surfaces is proposed. This algorithm is based on local surface fitting with a cubic triangular Bézier patch. The method has no requirement on how the neighbor points of the target point are distributed and has no requirement on the minimum number of neighbor points to be used.

The triangular Bézier patch is a special kind of Bézier surface [Far86, Far93], whose control points form a triangular net. The control point structure of a cubic triangular Bézier patch is:

$$\begin{array}{ccccccc} & & & \mathbf{P}_{003} & & & \\ & & \mathbf{P}_{102} & & \mathbf{P}_{012} & & \\ & \mathbf{P}_{201} & & \mathbf{P}_{111} & & \mathbf{P}_{021} & \\ \mathbf{P}_{300} & & \mathbf{P}_{210} & & \mathbf{P}_{120} & & \mathbf{P}_{030} \end{array} \tag{7.1}$$

and a triangular Bézier patch of degree n can be expressed as [Far86]:

$$\mathbf{s}(u,v) = \sum_{i,j,k\geq 0} \mathbf{P}_{i,j,k} B^n_{i,j,k}(u,v,w) \tag{7.2}$$

where $i+j+k=n$, $u+v+w=1$, u, v, and w are local barycentric coordinates. The basis functions $B^n_{i,j,k}$ are Bernstein polynomials of degree n:

$$B^n_{i,j,k}(u,v,w) = \frac{n!}{i!\,j!\,k!} u^i v^j w^k \tag{7.3}$$

Cubic Bernstein polynomials can be arranged in the following triangular scheme:

$$\begin{array}{ccccccc} & & & v^3 & & & \\ & & 3v^2w & & 3uv^2 & & \\ & 3vw^2 & & 6uvw & & 3u^2v & \\ w^3 & & 3uw^2 & & 3u^2w & & u^3 \end{array} \tag{7.4}$$

First the statement how to obtain the 10 control points $\mathbf{P}_{ijk}$ of a cubic triangular Bézier patch has to be made. Consider one triangle with vertices $\{\mathbf{V}_i, \mathbf{V}_j, \mathbf{V}_k\}$ in the triangulated

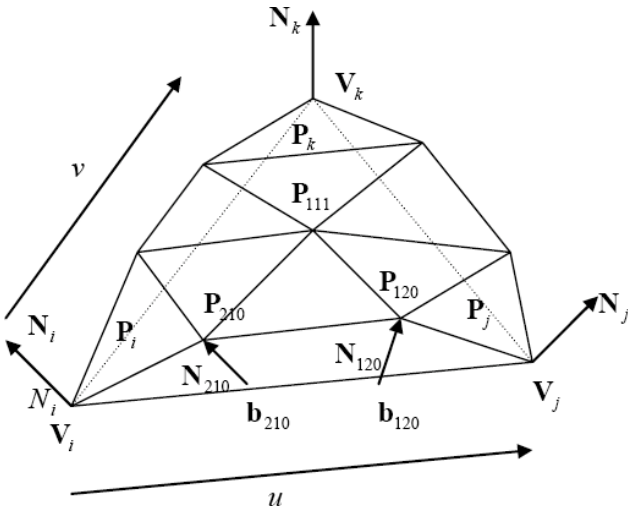


Figure 7.2.1: The control points of a cubic triangular patch.

surface. Equating to the triangular net introduced above, one gets: $\mathbf{P}_{300} = \mathbf{V}_i$, $\mathbf{P}_{030} = \mathbf{V}_j$, and $\mathbf{P}_{003} = \mathbf{V}_k$.

The normals $\{\mathbf{N}_i, \mathbf{N}_j, \mathbf{N}_k\}$ are the normals of the vertices $\{\mathbf{V}_i, \mathbf{V}_j, \mathbf{V}_k\}$, respectively. Each normal is the average of the normals of the triangles around the shared vertex. Each normal and its associated vertex define a tangent plane $\{\mathbf{P}_i$, $\mathbf{P}_j$, $\mathbf{P}_k\}$, respectively, perpendicular to the normal and across the vertex. The normal of a triangle can be obtained by a cross product of two edges of the triangle. First let us compute the control points on the edges and use edge $\overrightarrow{\mathbf{V}_i\mathbf{V}_j}$ as an example (see Figure 7.2.1). By an even division of the edge $\overrightarrow{\mathbf{V}_i\mathbf{V}_j}$ into three segments one obtains two points b_{210} and b_{120} :

$$\mathbf{b}_{210} = \frac{2}{3}\mathbf{V}_i + \frac{1}{3}\mathbf{V}_j \text{ and } \mathbf{b}_{120} = \frac{1}{3}\mathbf{V}_i + \frac{2}{3}\mathbf{V}_j \tag{7.5}$$

$\mathbf{b}_{210}$ and $\mathbf{b}_{120}$ have their normals defined respectively as:

$$\mathbf{N}_{210} = \frac{2}{3}\mathbf{N}_i + \frac{1}{3}\mathbf{N}_j \text{ and } \mathbf{N}_{120} = \frac{1}{3}\mathbf{N}_i + \frac{2}{3}\mathbf{N}_j \tag{7.6}$$

Then control points $\mathbf{P}_{210}$ and $\mathbf{P}_{120}$ are computed by respectively intersecting $\mathbf{N}_{210}$ and $\mathbf{N}_{120}$ with the tangent planes $\mathbf{P}_i$ and $\mathbf{P}_j$ from $\mathbf{b}_{210}$.

Since curvatures are functions of derivatives up to second order, it is intuitively sufficient to use second-order surfaces to estimate these differential parameters. Some experiments

have confirmed this assumption and show little advantage with 3rd and 4th-order surfaces over 2nd order surfaces [KLM98]. So the internal control point $\mathbf{P}_{111}$ is obtained by a method that maintains quadratic precision of the interpolation:

$$\mathbf{P}_{111} = \frac{1}{4}(\mathbf{P}_{201} + \mathbf{P}_{102} + \mathbf{P}_{021} + \mathbf{P}_{012} + \mathbf{P}_{210} + \mathbf{P}_{120}) - \frac{1}{6}(\mathbf{V}_i + \mathbf{V}_j + \mathbf{V}_k) \tag{7.7}$$

The above method for obtaining control points can guarantee the targeted vertex, $\mathbf{V}_i$, for example, shares the same tangent plane with its associated triangular Bézier patches at $\mathbf{V}_i$. Thus, these control points make a simple and good approximation of the triangulated surface that appears similar to a PN triangle [VPBM01], but a PN triangle targets better visual quality while the presented method targets better geometry approximation. Overveld and Wyvill [VOE97] present a detailed discussion on how to choose the control points from the view of surface subdivision.

A parameterized C'-surface is a C'-differentiable map $\mathbf{X} : Q \to E^3$ of an open domain $Q(u,v) \in R^2$ into the Euclidian space E^3 , where $\mathbf{X}_1 := \frac{\partial \mathbf{X}}{\partial u}$ and $\mathbf{X}_2 := \frac{\partial \mathbf{X}}{\partial w}$ are linearly independent. The two-dimensional linear subspace $T_p\mathbf{X}$ of E^3 generated by the span $\{\mathbf{X}_1, \mathbf{X}_2\}$ is called the tangent space of $\mathbf{X}$ at $\mathbf{p}$. The unit normal field $\mathbf{N}$ is given by

$$\mathbf{N} := \frac{[\mathbf{X}_1, \mathbf{X}_2]}{\|[\mathbf{X}_1, \mathbf{X}_2]\|}$$

where $[,] : E^3 \times E^3 \to E^3$ is the cross product.

The moving frame $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{N}\}$ is called the Gaussian Frame. The Gaussian Frame is in general not an orthogonal frame. The bilinear form on $T_p\mathbf{X}$ induced by the inner product of E^3 by restriction is called the first fundamental form of the surface. The matrix representation of the first fundamental form $\mathbf{I}_p$, with respect to the basis $\{\mathbf{X}_1, \mathbf{X}_2\}$ of $T_p\mathbf{X}$, is given by

$$\begin{bmatrix} g_{11} & g_{12} \\ g_{11} & g_{22} \end{bmatrix} = \begin{bmatrix} \langle \mathbf{X}_1, \mathbf{X}_1 \rangle & \langle \mathbf{X}_1, \mathbf{X}_2 \rangle \\ \langle \mathbf{X}_2, \mathbf{X}_1 \rangle & \langle \mathbf{X}_2, \mathbf{X}_2 \rangle \end{bmatrix} \tag{7.8}$$

where $\langle , \rangle : E^3 \cdot E^3 \to E$ is the dot product.

The first fundamental form $\mathbf{I}_p$ is symmetric, positive definite, and geometric invariant. Geometrically the first fundamental form allows measurements on the surface (length of curves, angles between tangent vectors, areas of regions) without referring back to the space E^3 in which the surface lies.

The linear map $\mathbf{L} : T_p\mathbf{X} \to T_p\mathbf{X}$ defined by $\mathbf{L} := -\frac{d\mathbf{N}}{d\mathbf{X}}$ is called the Weingarten map. The bilinear form $\mathbf{II}_p$ defined by $\mathbf{II}_p(\mathbf{A},\mathbf{B}) := \langle \mathbf{L}(\mathbf{A}),\mathbf{B}\rangle$ for each $\mathbf{A},\mathbf{B} \in T_p\mathbf{X}$ is called the second fundamental form of the surface. The matrix representation of $\mathbf{II}_p$, with respect to the basis $\{\mathbf{X}_1,\mathbf{X}_2\}$ of $T_p\mathbf{X}$, is given by

$$h_{ij} := \langle -\mathbf{N}_i, \mathbf{X}_j\rangle = \langle \mathbf{N}, \mathbf{X}_{ij}\rangle \; i,j = 1,2 \tag{7.9}$$

The Weingarten map $\mathbf{L}$ is self-adjoint. The eigenvalues k_1 and k_2 are therefore real and the corresponding eigenvectors are orthogonal. The eigenvalues k_1 and k_2 are called the principal curvatures of the surface.

$$K := k_1 \cdot k_2 = \det(\mathbf{L}) = \frac{\det(\mathbf{II})}{\det(\mathbf{I})} \tag{7.10}$$

is called the Gaussian curvature and

$$h_{ij} := \langle -\mathbf{N}_i, \mathbf{X}_j\rangle = \langle \mathbf{N}, \mathbf{X}_{ij}\rangle \; i,j = 1,2 \tag{7.11}$$

is called the mean curvature.

Given equation (7.10), one needs to obtain the first fundamental form $(\mathbf{I}_p)$ and second fundamental $(\mathbf{II}_p)$ form of the parameterized surface, namely the cubic triangular Bézier patch. From equations (7.2) and (7.3), the first and second derivatives of Bernstein polynomials of degree 3 have to be computed. The triangular scheme for cubic Bernstein polynomials in equation (7.4) is very useful to express the result.

Considering $u+v+w=1$, one has $\frac{\partial w}{\partial u} = \frac{\partial w}{\partial v} = -1$. So for the first and second derivatives of 3rd degree Bernstein polynomials at location $u=0, v=0$, i.e., at the vertex $\mathbf{V}_i$, one has:

$$
\begin{array}{ll}
\left.\dfrac{\partial B^3_{i,j,k}(u,v,w)}{\partial u}\right|_{u=0,v=0,w=1} & \begin{array}{cccc} & & 0 & \\ & 0 & & 0 \\ 0 & & 0 & & 0 \\ -3 & & 3 & & 0 & & 0 \end{array} \\
\left.\dfrac{\partial B^3_{i,j,k}(u,v,w)}{\partial v}\right|_{u=0,v=0,w=1} & \begin{array}{cccc} & & 0 & \\ & 0 & & 0 \\ 3 & & 0 & & 0 \\ -3 & & 0 & & 0 & & 0 \end{array} \\
\left.\dfrac{\partial^2 B^3_{i,j,k}(u,v,w)}{\partial u^2}\right|_{u=0,v=0,w=1} & \begin{array}{cccc} & & 0 & \\ & 0 & & 0 \\ 0 & & 0 & & 0 \\ 6 & & -12 & & 6 & & 0 \end{array} \\
\left.\dfrac{\partial^2 B^3_{i,j,k}(u,v,w)}{\partial v^2}\right|_{u=0,v=0,w=1} & \begin{array}{cccc} & & 0 & \\ & 6 & & 0 \\ -12 & & 0 & & 0 \\ 6 & & 0 & & 0 & & 0 \end{array} \\
\left.\dfrac{\partial^2 B^3_{i,j,k}(u,v,w)}{\partial u \partial v}\right|_{u=0,v=0,w=1} & \begin{array}{cccc} & & 0 & \\ & 0 & & 0 \\ -6 & & 6 & & 0 \\ 6 & & -6 & & 0 & & 0 \end{array}
\end{array}
\tag{7.12}
$$

So for cubic triangular Bézier patch $\mathbf{s}(u,v) = \sum_{i,j,k\geq 0} \mathbf{p}_{i,j,k} B^3_{i,j,k}(u,v,w)$ it holds:

$$
\begin{aligned}
\mathbf{X}_1 \equiv \left.\frac{\partial \mathbf{s}(u,v)}{\partial u}\right|_{u=0,v=0} &= 3\cdot\mathbf{P}_{210} - 3\cdot\mathbf{P}_{300} \\
\mathbf{X}_2 \equiv \left.\frac{\partial \mathbf{s}(u,v)}{\partial v}\right|_{u=0,v=0} &= 3\cdot\mathbf{P}_{201} - 3\cdot\mathbf{P}_{300} \\
\mathbf{X}_{11} \equiv \left.\frac{\partial^2 \mathbf{s}(u,v)}{\partial u^2}\right|_{u=0,v=0} &= 6\cdot\mathbf{P}_{120} - 12\cdot\mathbf{P}_{210} + 6\cdot\mathbf{P}_{300} \\
\mathbf{X}_{22} \equiv \left.\frac{\partial^2 \mathbf{s}(u,v)}{\partial v^2}\right|_{u=0,v=0} &= 6\cdot\mathbf{P}_{102} - 12\cdot\mathbf{P}_{201} + 6\cdot\mathbf{P}_{300}
\end{aligned}
$$

$$\mathbf{X}_{12} \equiv \left.\frac{\partial^2 \mathbf{s}(u,v)}{\partial u \partial v}\right|_{u=0,v=0} = 6\cdot\mathbf{P}_{111} - 6\cdot\mathbf{P}_{210} - 6\cdot\mathbf{P}_{201} + 6\cdot\mathbf{P}_{300} \tag{7.13}$$

From equation 7.8 one has the first fundamental form of $X(u,v)$ at the position $u = v = 0$:

$$g_{ij} = \langle \mathbf{X}_i, \mathbf{X}_j \rangle, \qquad i,j = 1,2$$

Suppose $\mathbf{N}$ is the surface normal of $X(u,v)$ at the position $u = v = 0$. From equation 7.9, one has the second fundamental form of $X(u,v)$ at the position $u = v = 0$:

$$h_{ij} = \langle \mathbf{N}, \mathbf{X}_{ij} \rangle, \qquad i,j = 1,2$$

For Gaussian and mean curvature at the position $u = v = 0$ one has:

$$\begin{aligned} K_T &= \frac{h_{11}h_{22} - h_{12}^2}{g_{11}g_{22} - g_{12}^2} \\ H_T &= \frac{g_{22}h_{11} - 2g_{12}h_{12} + g_{11}h_{22}}{2(g_{11}g_{22} - g_{12}^2)} \end{aligned} \tag{7.14}$$

The next step is to come up with an averaging scheme. As shown in [MDSB02] the averaging using Voronoi regions gives the smallest error. For a non-obtuse triangle $(\mathbf{V}_i, \mathbf{V}_j, \mathbf{V}_k)$ the Voronoi area can be calculated with the cotangent formula as:

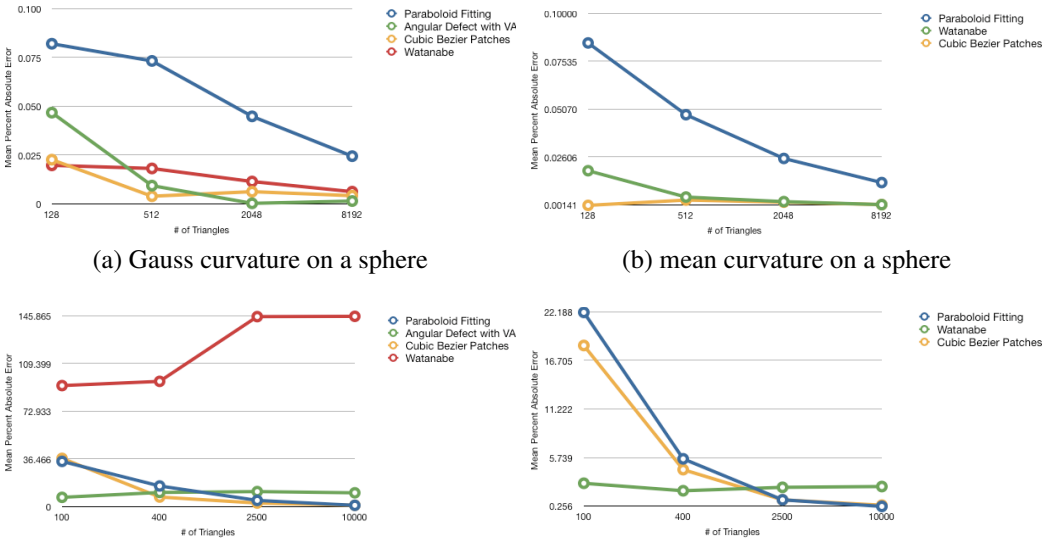
$$d_r = \cot\alpha_{ij}\|\mathbf{V}_i - \mathbf{V}_j\|^2 + \cot\beta_{ik}\|\mathbf{V}_k - \mathbf{V}_i\|^2, \tag{7.15}$$

where α_{ij} and β_{ik} are the angles at $\mathbf{V}_k$ and $\mathbf{V}_j$ respectively.

The curvatures are calculated as following:

$$\begin{aligned} D_r &= \frac{d_r}{d_1 + d_2 + \ldots + d_m} \\ K(\mathbf{V}_i) &= D_1K_1 + D_2K_2 + \ldots + D_mK_m \\ H(\mathbf{V}_i) &= D_1H_1 + D_2H_2 + \ldots + D_mH_m \end{aligned} \tag{7.16}$$

where $d_1 \ldots d_m$ are the Voronoi areas of all adjacent triangles from 1-ring neighbourhood.



(a) Gauss curvature on a sphere

(b) mean curvature on a sphere

(c) Gauss curvature on a torus

(d) mean curvature on a torus

Figure 7.2.2: Comparison results of the presented method

This gives us the principle curvatures at Vertex $\mathbf{V}_i$

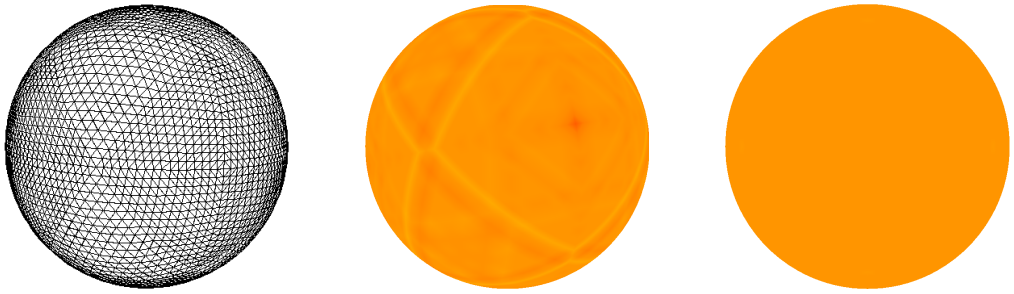
$$\begin{aligned} \kappa_1(\mathbf{V}_i) &= H(\mathbf{V}_i) + \sqrt{H(\mathbf{V}_i)^2 - K(\mathbf{V}_i)} \\ \kappa_2(\mathbf{V}_i) &= H(\mathbf{V}_i) - \sqrt{H(\mathbf{V}_i)^2 - K(\mathbf{V}_i)} \end{aligned} \tag{7.17}$$

7.2.3 Results and applications

This section compares the proposed algorithm with the compelling methods for curvature estimation. The measure which is used for this comparison is the mean percent absolute error with respect to the analytically calculated values of curvature. The method of generalized focal surface is also used to visually represent the calculated curvature and it will be shown later that it can help in assessing the quality of the tesselation.

7.2.4 Comparison to other methods

Some experiments with the presented curvature-estimation algorithm were conducted to demonstrate the accuracy of the method in practice. Simple shape models such as a sphere



(a) Triangulated surface (b) Cubic Bézier Patches method (c) Angular Defect method

Figure 7.2.3: Triangulation and Gauss curvature on a unit sphere. Triangulation is obtained by recursive subdivision with five subdivision steps. Gauss curvature color coding: red color indicates the highest percentual difference from the analytical values.

and a torus were used as the test datasets, because the curvatures of them are known analytically, which simplifies the comparison to the ground truth. According to comparisons made in [MSR07] the two best methods for mean and Gaussian curvature estimation are paraboloid fitting and Gauss-Bonnet scheme. However, the experiments show that angular defect algorithm using Voronoi areas averaging scheme gives better approximation for irregular meshes. The probable importance of this fact depends on the application area of the curvature estimation method. For the experiments three methods were chosen: paraboloid fitting, Gauss-Bonnet and Watanabe algorithms. Experimental results for the proposed scheme and these three competing methods are shown in Figure 7.2.2a, 7.2.2b, 7.2.2c and 7.2.2d.

The motivation of the proposed algorithm is to construct a set of triangular Bézier patches for local surface fitting. The curvature of each patch is accurately calculated and the area-based average of these curvatures gives the estimation of curvature at the specific location. Consequently, the accuracies of both Gaussian and mean curvatures are of the same order, and the calculation is robust from an implementation standpoint.

The effect of the irregularity of the mesh also was tested. A mesh on a unit sphere (see Figure 7.2.3a) was generated by recursive subdivision of an octagon centered at the origin with the position of each vertex normalized after each iteration. Ideally speaking, since the parameters of a sphere are identical, the mesh on a sphere should be regularly distributed. However, the mesh obtained by recursive subdivision is irregularly distributed, especially at the locations of the original boundaries of the octagon. Since the triangles of this

test data are non-obtuse triangles, discrete differential-geometry operators worked rather well and showed little difference in the irregular areas (see Figure 7.2.3c). However, the behavior off the presented algorithm is different: the algorithm gives varying values in the irregular areas and almost constant results in the regular areas (see Figure 7.2.3b). Such behavior is reasonable and useful, because for surface interrogation, one does not only need to estimate the differential-geometry of the approximated surfaces, but also the quality of the tessellation which is essential for such applications as simulation.

The construction of the control points is based on the normal vectors at the vertices of the triangulated surfaces. The proposed algorithm yields satisfactory results by simply adopting an even-weighted method to estimate the normal vectors. Even with more sophisticated methods of normal-vector estimation, its behavior on the irregular surface areas remains the same.

Besides numerical comparisons of the proposed technique, visual results are given below to show how these algorithms can be applied to different visualization areas. The presented curvature estimation algorithm is used in a generalized focal surface method to show quantities of several ideally parameterized surfaces.

7.2.5 Visual results on a generalized focal surface

Focal surfaces are based on line congruencies [Far93]. Line congruencies can be used to visualize any scalar on a surface. Particular generalized focal surfaces can be very useful tools in surface interrogation as was first shown in [HPD91]. Focal surfaces can be useful to analyze the quality of a surface before further processing the surface, as in a numerically controlled milling operation, for example.

The surfaces can be represented parametrically as vector-valued functions $\mathbf{X}(u,w)$, with $\mathbf{N}(u,w)$ as the unit normal vector of the surface.

$$\mathbf{F}_i(u,w) = \mathbf{X}(u,w) + \kappa_i^{-1}(u,w)\mathbf{N}(u,w), \quad i = 1,2 \tag{7.18}$$

is then the parametric representation of the focal surface, with κ_1 and κ_2 being the principal curvatures of the surface.

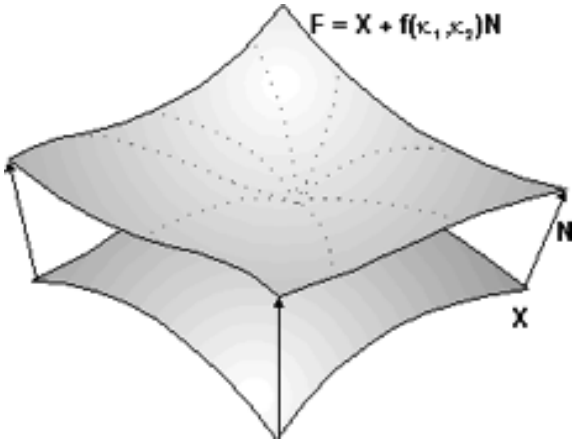


Figure 7.2.4: A generalized focal surface.

Hagen and Hahmann in [HPD91] generalized the classical focal surface concept to achieve a new surface interrogation tool:

$$\mathbf{F}(u,w) := \mathbf{X}(u,w) + a \cdot f(\kappa_1, \kappa_2)\mathbf{N}(u,w), \tag{7.19}$$

where the scalar function $f(\kappa_1, \kappa_2)$ now depends on the principal curvatures $\kappa_1(u,w)$ and $\kappa_2(u,w)$. This scalar function is called the interrogation function. A schematic representation of this type of focal surface is shown in Figure 7.2.4.

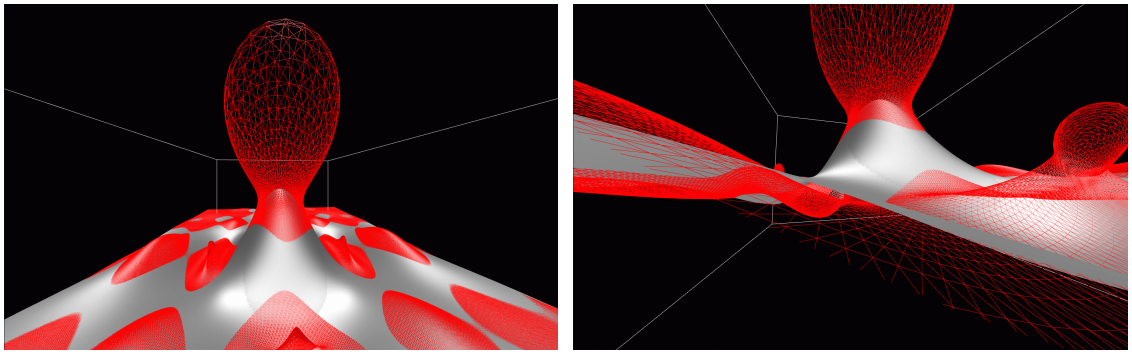
The real number a is used as a scale factor. If the curvatures are very small you need a very large number a to distinguish the two surfaces $\mathbf{X}(u,w)$ and $\mathbf{F}(u,w)$ on the screen. Varying this factor can also improve the visibility of several properties of the focal surface. For example, by varying a, one may be able to see intersections or small details more clearly.

Different interrogation functions $f(\kappa_1, \kappa_2)$ are used for different applications:

- Convexity test: $f = \kappa_1 \cdot \kappa_2$ (Gaussian curvature),
- Continuity test: $f = \kappa_1 + \kappa_2$.

Different offset functions can be used to interrogate and visualize surfaces with respect to the following criteria:

- detection of flat points,



(a) (b)

Figure 7.2.5: Convexity test for the surface $z = \frac{\sin(x)\sin(y)}{xy}$ using the interrogation function $f(\kappa_1, \kappa_2) = \kappa_1 \cdot \kappa_2$. The solid gray surface is the target surface while the red meshed surface is the generalized focal surface. The right view clearly shows the intersection between the surface $z = \frac{\sin(x)\sin(y)}{xy}$ and its focal surface.

- detection of surface irregularities,
- visualization of curvature behavior,
- visualization of technical smoothness,
- visualization of C^2 and C^3 discontinuities.

For more details, see [HPD91].

To demonstrate the quality of the presented method, the convexity test of a triangulated surface of function $z = \frac{\sin(x)\sin(y)}{xy}$ using the interrogation function $f(\kappa_1, \kappa_2)$ is displayed in Figure 7.2.5. Figure 7.2.6 shows the continuity test of a race car body with the interrogation function $f(\kappa_1, \kappa_2) = \kappa_1^2 + \kappa_2^2$.

In the convexity test, the intersection of the surface with its focal surface indicates the line of zero Gaussian curvature. The formula for the variable offset factor used in the convexity test is $f = \kappa_1 \cdot \kappa_2$. In Figure 7.2.5, the intersections between the interrogated surface and its focal surface can be observed.

In the continuity test, the discontinuity of the focal surface indicates curvature discontinuity. The formula for the variable offset factor used in the continuity test is $f = \kappa_1^2 + \kappa_2^2$. The discontinuity of the focal surface indicates the discontinuity of the surface in that area.

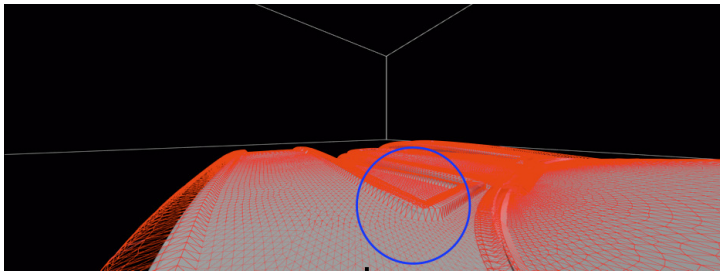


Figure 7.2.6: Continuity test for a race car body using the interrogation function $f(\kappa_1, \kappa_2) = \kappa_1^2 + \kappa_2^2$. The area circled in blue shows sharp changes in the focal surface, which indicate the discontinuity of the surface in that area. The solid gray surface is the target surface and the red mesh is the generalized focal surface.

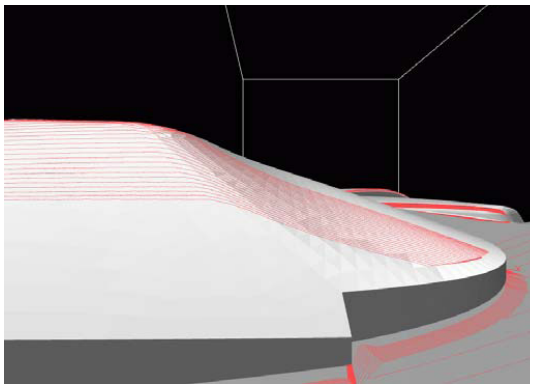


Figure 7.3.1: Reflection lines of C^0 continuous surface.

In Figure 7.2.6 sharp changes of the focal surface can be easily observed, for example, in the blue circle area.

7.3 Reflections lines for tesselation quality assessment

7.3.1 Theoretical background and state of the art

The reflection lines method is a surface interrogation method based on curves over a surface [HGM02, KK88]. The reflection line method determines unwanted curvature regions by irregularities in the reflection line pattern of parallel light lines. If the interrogated surface is C^r continuous, the interrogation lines will be C^{r-1} continuous. It is difficult to visualize and detect the difference between C^2 continuity and C^1 continuity for the interro-

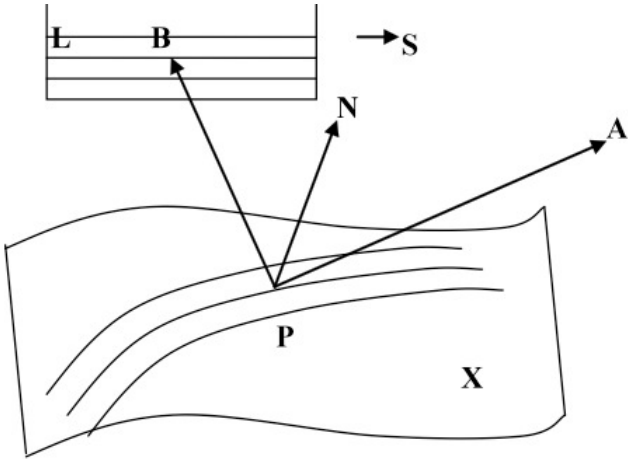


Figure 7.3.2: Reflection lines on a surface. **L** is a light line. **B** is a point on **L**. **N** is the surface normal. **P** is the reflected point of point **B** on the surface. **A** is the eye point. **X** is the surface. **S** is the direction of **L**.

gation lines on a shaded surface. However, C^0 continuity for the interrogation lines means one is looking for corners in the computed lines, something one can clearly observe. For C^0 continuous areas of the surface, the interrogation lines appear discontinuous. In Figure 7.3.1, twenty-five reflection lines on a car model are rendered. As expected, one can easily see the discontinuity in the reflection lines in the right-bottom corner (blue circle), where the interrogated surface is only C^0.

A so-called light line **L** is given in parameter form:

$$\mathbf{L}(t) = \mathbf{L}_0 + t \cdot \mathbf{S}, \qquad t \in \mathbb{R}$$

where **S** is the light line direction.

The reflection line is the projection of the line **L** on the surface **X** that can be seen from the eye point **A** (if the light line **L** is reflected on the surface).

From the geometric dependencies shown in Figure 7.3.2, the following reflection condition can be easily derived:

$$\frac{\mathbf{a}}{\|\mathbf{a}\|} + \frac{\mathbf{b}}{\|\mathbf{b}\|} = 2\mathbf{N}\langle \mathbf{N}, \frac{\mathbf{b}}{\|\mathbf{b}\|} \rangle = 2\mathbf{N}\langle \mathbf{N}, \frac{\mathbf{a}}{\|\mathbf{a}\|} \rangle,$$

where $\mathbf{a} = \mathbf{P} - \mathbf{A}$ and $\mathbf{b} = \mathbf{B} - \mathbf{P}$ and **B** is a point on **L**.

The parallel set of reflection lines in direction **S** is now mapped from the "light plane" on to the surface (called "mirroring"). Stepping along each curve of the setfor a fixed eye point one obtains a non-linear system of equations for the unknown parameters **a** and **b**:

$$\mathbf{b} + \lambda \cdot \mathbf{a} = 2\mathbf{N}\langle \mathbf{N}, \mathbf{b} \rangle, \qquad \lambda = \frac{\|\mathbf{b}\|}{\|\mathbf{a}\|}$$

These three non-linear equations can be reduced to two equations by eliminating λ and this system can be solved by numerical methods. For free-form surfaces, the modified Newton method is widely used to solve this non-linear problem, but the existence and unambiguousness of the solution has to be ensured by an appropriate position of the eye point. For concave surfaces, the solution for a reflection line is not unique.

The characteristics and rendering techniques of reflection lines have been studied by many researchers [Ham93, LGS99, KLPK01, SGG02, TF97]. Heckbert and Hanrahan tried to speed up reflection computation by sweeping areas through a scene to form "beams" [Ham93]. Loos, Greiner, and Seidel developed a minimization algorithm based on data dependent quadratic error functionals to enable the designer to control during the modeling process the reflection line pattern by approximating a specified family of reflection lines [LGS99]. Sussner, Greiner, and Grosso presented an algorithm for the tessellation of trimmed surfaces that generates a 2-manifold mesh with a low triangle count and a triangulation pattern suitable for visualization [SGG02]. Theisel, and Farin analyzed the curvature of some characteristic curves and surfaces, such as contour lines, lines of curvatures, asymptotic lines, isophotes and reflection lines [TF97].

In triangulated surfaces, the vertices on the surface are sampled irregularly. There is no reference to a uniform parameter space as with free form surfaces. It is even harder for the modified Newton method to compute reflection lines on triangulated surfaces because it is difficult to accurately estimate the derivatives of a triangulated surface. However, derivatives are critical in the modified Newton method.

A graphical approach for rendering reflection lines over triangulated surface is to utilize reflection mapping [BN76], which can be accomplished in real time on modern graphics cards. However, this approach has to adopt parallel light cylinders instead of parallel light lines to avoid serious sampling problems, especially at the locations of high curva-

ture. Experiment results have confirmed such phenomena [BMG03]. Thus, this graphical approach reduces the spatial accuracy.

To accurately compute reflection lines on triangulated surface, a simple solution is contouring:

1. For each vertex on the triangulated surface, such as the $\mathbf{V}_1$ in Figure 7.3.3, compute the reflection ray of the ray from the eye position to this vertex, i.e., the reflection ray $\overrightarrow{\mathbf{V}_1\mathbf{V}_1'}$ of the ray from the eye position to $\mathbf{V}_1$;
2. For each vertex, compute the distance between the reflection ray and one of the light lines;
3. Compute contour lines on these distances with threshold of zero, then these contour lines are the reflection lines associated with the light line;
4. Change to another light line and repeat steps 2 and 3 to compute the reflection lines for all light lines.

This contouring algorithm is stable and overcomes the difficulties of the modified Newton method, namely the existence and the unambiguousness of the solution. However, its computational cost is still too high for interactive applications: for reflection lines of each light line, step 2 – the computation of the distance between two three-dimensional lines – must be performed for the reflection rays of all vertices, regardless whether the vertex can be seen or not.

7.3.2 A method for reflection lines computation

To overcome the disadvantages of the contouring algorithm described in the previous section, in this section a novel approach for reflection line computation on a triangulated surface is proposed. The method maps the triangulation onto the light plane (see Figure 7.3.3).

For each vertex on the triangulated surface,

- compute its reflection ray;

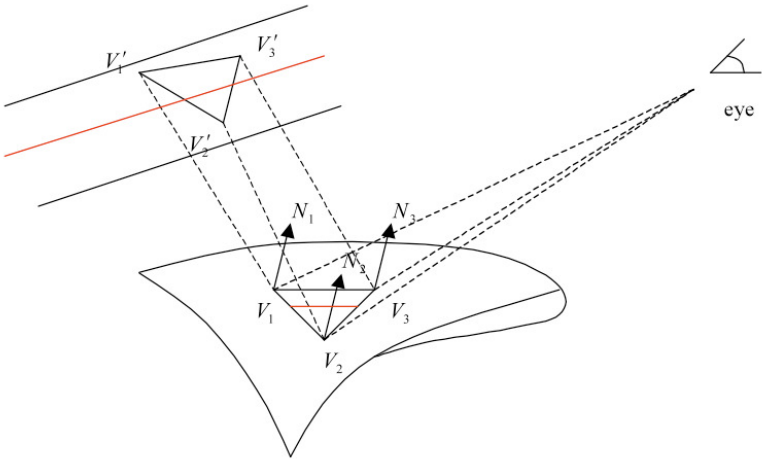


Figure 7.3.3: A schematic diagram showing $\mathbf{V}_i$'s and $\mathbf{N}_i$'s used in computing reflection lines on a triangulated surface.

- test if this *directional* ray intersects the light plane to determine which triangles are seen; if an intersection exists, compute its location;

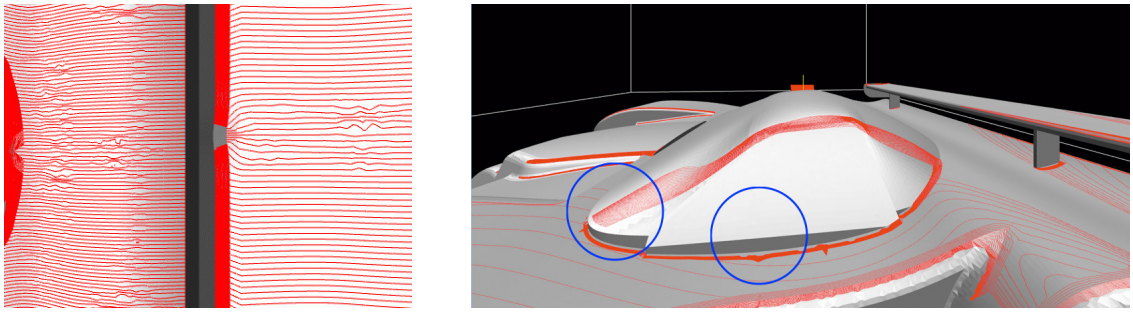
- because step 2 has mapped the three-dimensional triangulated surface into the two dimensional light plane, the intersections of the light lines with these triangles can be computed;

- map the intersections obtained in step 3 "down" to the original triangulated surface; these intersections form the final result – reflection lines of all the light lines in the light plane.

The light lines in the light plane are parallel and usually have a very simple formula, such as,

$$x = k \cdot a, \quad a = 0.1, \quad k = \ldots, -2, -1, 0, 1, 2, \ldots, \quad z = 1$$

Thus, it is very easy to compute the intersections between the light lines and the mapped triangles in the light plane in the step 3. The intersection distance along the edges of the triangles can be expressed easily from simple plus-minus calculations of the coordinates of the vertices and parameters of the light lines and can be used for the "mapping-down" procedure in step 4. The high cost for computation of the distance between reflection ray and light line is alleviated. Because only visible triangles have been mapped onto the light



(a) Zoomed in top view

(b) Side view. Blue circles: surface irregularities and discontinuity

Figure 7.3.4: Reflection lines on a surface of a race car

plane in step 2 for later computation of the reflection lines, about half of the computations have been saved.

7.3.3 Results and discussion

Figure 7.3.4a shows part of the surface of a race car model with 500 reflection lines emanating from the whole surface. The complete surface contains 400,000 triangles. Figure 7.3.4a is a view from above the car; the light plane is above the car as well. Figure 7.3.4a shows how reflection lines look when they densely cover the interrogated surface. In Figure 7.3.4a, some small areas show distortions of the reflection lines, which indicate the existence of surface irregularities (in the blue circled areas). The aliasing of the reflection lines observable in the left part of Figure 7.3.4a is due to the resolution limitation of the screen or printing process. Figure 7.3.4b shows side views of the same car with 10 reflection lines; irregularity on the surface (middle blue circle area) and discontinuity of reflection lines (left blue circle area) can be clearly observed.

7.4 Summary

This chapter presented two novel approaches for surface interrogation of triangulated surfaces. These two approaches have been designed to overcome some limitations of the traditional methods. The proposed method for curvature estimation uses a simple com-

putation of the derivatives of a free-form patch on each triangle. These derivatives can be derived by applying arithmetic operations on the control points of the free-form patch. Compared with other curvature estimation algorithms, the approach removes the requirements on a minimum number of neighboring points and the way they are distributed. The presented algorithm for the computation of reflection lines avoids the computation of the distance between the reflection ray and the light line in three dimensions. In the traditional contouring algorithm this procedure has to be repeated for each light line, which influences the overall performance. The proposed algorithm naturally obtains an alternative scalar measurement for all light lines in a single step by mapping the triangulated surface onto the light plane. Further improvement can be made by ignoring the triangles hidden from the viewer. The presented approaches make it possible to concurrently visualize results from simulation and surface interrogation and to reveal the possible intrinsic relationship between them.

Chapter 8

Conclusion and Future Work

Scalar field data comes from measurements and simulations in different areas. The visual representation of it has become an important and intuitive way for data exploration and interpretation. With the technical improvements on the data acquisition side, resulting in larger more densely sampled datasets, the requirements on the visualization methods grow. Performance, accuracy, visual esthetic quality, new ways of extracting knowledge from data are examples of the areas demanding an active research. This thesis discussed several applications of computational topology to the visualization of scalar fields, addressing some of the mentioned demands.

Volume rendering is the most common technique for the visualization of 3D scalar fields nowadays. It gives a realistic picture of the data by approximating the volume rendering integral (see Section 2.2.2) and allows the user interactively manipulate the visualization by adjusting the transfer function. Because of these advantages volume rendering is used in a wide range of applications. Basically every system visualizing any 3D real-world data utilizes some kind of a volume rendering technique. Due to the large size of the datasets and the flexibility required current techniques approximating the volume rendering integral are computationally expensive, depending on the data and the chosen transfer function. Adjustable parameters such as sampling rate along the ray, give the possibility for a tradeoff between quality and interactivity. It is a common situation that increasing the rendering speed means introducing more rendering artifacts into the scene. Therefore improvements on the volume rendering techniques, which are beneficial for some

special kinds of applications, allowing to optimize the mentioned tradeoff under some circumstances are required. One of such improvements, multidimensional peak finding, was introduced in this work. The method is specifically designed for noisy data and high frequency transfer functions, which is usually the case for isosurface rendering. A performance and visual quality comparison with the existing methods shows clear benefits of the technique. A presented improvement on the method which utilizes discrete Morse theory helps to increase performance with a little memory overhead. Multidimensional transfer functions are a powerful tool for some problems, though interactive specification of such a transfer function is not a trivial task. To allow the user real time transfer function manipulation one needs a system with a corresponding editing widget. Several attempts to create such a widget are implemented in known rendering systems, though they are neither flexible nor intuitive to use. One of the directions for future work might be providing of such an editing tool to allow the user for convenient work and parameter manipulation with a presented volume rendering method. Another direction for future research would be exploring combinatorial topological methods for segmentation in presence of degenerate regions. Filtering was presented here as a solution for this problem, other ideas to try are, for example, artificial noise or tilting the regression plane of a 2D manifold.

With the development of the topological methods for large and noisy data, topological simplification got much attention. Nowadays one can not imagine a topological structure without being able to simplify it. Simplification aims at reducing the feature space and is nothing else than removing ”unimportant” features. In the terms of Morse theory features are represented by critical points. The process of removing them is called cancellation and is performed pairwise on the critical points with the lowest importance value. An importance measure commonly used nowadays, homological persistence, has a clear drawback in the presence of outlier-like noise in the data: the outliers get high persistence values and therefore are not being removed. In this thesis a new importance measure for critical points in a 2D scalar field was presented. It is called scale space persistence. In the presence of outlier-like noise in the data it is clearly beneficial in comparison with a commonly known homological persistence measure. The computation of the scale space persistence utilizes a concept of scale space from the area of computer vision. Scale space gives the information of the spatial extent of a critical point, how ”wide is the hill around it”. Using this information, one can easily sort out the outliers, which have small spatial

extent, but are highly important in the sense of homological persistence. Chapter 5 also presented a real world application of the method, the crater identification on the surface of Mars. The presented algorithm is scalable, requires no preprocessing and can be implemented in parallel. As a direction for future work, one can investigate the application of non-linear scale spaces which can be easily integrated into scale space persistence measure. Also one can have a closer look at the multidimensional persistence theory [CZ09] with the purpose to incorporate the structural information contained in the scale space with the homological persistence concept.

Application of topology for visualization often requires deep understanding of the mathematical background behind it. Therefore topological methods are often considered to be complicated and hard to understand. A usual domain expert exploring the data using topology for feature extraction needs an intuitive way to manipulate the exploration process. In this thesis a system for general interactive topology analysis and exploration was presented. This system is based on an intuitive notion of a scene graph, where the user can choose and place the component blocks to achieve an individual result. The tool gives the possibility for calculation and simplification of the underlying topological structure, Morse-Smale complex, and also the visualization of parts of it. The system also includes a simple generic query language to acquire different structures of the topological structure at different levels of hierarchy. This work also discussed the data structures to maintain the simplification hierarchy and flexible rendering of features of different dimensions. Using the presented system the domain expert can extract more knowledge from given data independent on the application domain. The system presented in this thesis is a minimalistic set of functions for topology exploration and visualization using three-dimensional Morse-Smale complex structure. One of the directions for future work would be exploring topological structures other than Morse-Smale complex to make the tool more universal. The system can also profit from optimizations on the rendering pipeline and on data structures for more efficient storage and visualization.

Chapter 7 presented an early work of the author in the area of computational geometry, motivated by the need of new techniques for quality assessment of a triangulated surface, surface interrogation. Two techniques were introduced: a curvature estimation algorithm and a method for reflection lines computation. The algorithms were compared to the compelling methods and the results on real geometric models like a body of a race car were

presented. With these approaches one has the possibility to concurrently visualize results from simulation and surface interrogation and to reveal the possible intrinsic relationship between them.

Bibliography

[Arm10] A. Armstrong. *Basic Topology*. Undergraduate Texts in Mathematics. Springer, 2010.

[AT05] Gady Agam and Xiaojing Tang. A sampling framework for accurate curvature estimation in discrete surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 11(5):573–583, September 2005.

[AWC10] M. Ament, D. Weiskopf, and H. Carr. Direct Interval Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1505–1514, 2010.

[BCM03] V. Borrelli, F. Cazals, and J.-M. Morvan. On the angular defect of triangulations and the pointwise approximation of curvatures. *Computer Aided Geometric Design*, 20(6):319–341, September 2003.

[BEHP04] Peer-Timo Bremer, Herbert Edelsbrunner, Bernd Hamann, and Valerio Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):385–396, 2004.

[BLW10] Ulrich Bauer, Carsten Lange, and Max Wardetzky. Optimal topological simplification of discrete functions on surfaces. *CoRR*, abs/1001.1269, 2010.

[BMG03] M. E. Brill, R. J. Moorhead, and Y. Guan. Immersive surface interrogation. *Technical Report MSSU-COE-ERC-03-04*, April 2003.

[BMWM06] Steven Bergner, Torsten Moeller, Daniel Weiskopf, and David J. Muraki. A spectral analysis of function composition and its implications for sampling in direct volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12:2006, 2006.

[BN76] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *SIGGRAPH Computer Graphics*, 10(2):266–266, July 1976.

[BP07] Peer-Timo Bremer and Valerio Pascucci. A practical approach to two-dimensional scalar topology. In *Topology-based Methods in Visualization*, Mathematics and Visualization, pages 151–169. Springer Berlin Heidelberg, 2007.

[BPaLDCS06] Fábio F. Bernardon, Christian A. Pagot, Jo ao Luiz Dihl Comba, and Cláudio T. Silva. Gpu-based tiled ray casting using depth peeling. *Journal of Graphics Tools*, 11.3(ISSN 1086-7651):23–29, 2006.

[Bre65] J.E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.

[BWP+10] P.-T. Bremer, G. Weber, V. Pascucci, M. Day, and J. Bell. Analyzing and tracking burning structures in lean premixed hydrogen flames. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):248–260, 2010.

[Cay59] A. Cayley. On contour and slope lines. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, XVII:264–268, 1859.

[CCF94] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated Volume Rendering and Tomographic Reconstruction using Texture Mapping Hardware. In *VVS '94: Proceedings of the 1994 Symposium on Volume Visualization*, pages 91–98, New York, NY, USA, 1994. ACM Press.

[CCL03] F. Cazals, F. Chazal, and T. Lewiner. Molecular shape analysis based upon the Morse-Smale complex and the Connolly function. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, SCG '03, pages 351–360, New York, NY, USA, 2003. ACM.

[CCM+00] P. Cignoni, D. Constanza, C. Montani, C. Rocchini, and R. Scopigno. Simplification of tetrahedral meshes with accurate error evaluation. In *Proceedings of the Conference on Visualization '00*, VIS '00, pages 85–92, Los Alamitos, CA, USA, 2000. IEEE Computer Society Press.

[CDP+96] Tom Canfield, Terry Disz, Michael E. Papka, Rick Stevens, Milana Huang, Valerie Taylor, and Jian Chen. Toward real-time interactive virtual prototyping of mechanical systems: Experiences coupling virtual

reality with finite element analysis. *Proceedings of High Performance Computing*, pages 339–345, 1996.

[CL91] Nam Ik Cho and San Uk Lee. Fast algorithm and implementation of 2-d discrete cosine transform. *Circuits and Systems, IEEE Transactions on*, 38(3):297 –305, March 1991.

[CL03] Y. Chiang and X. Lu. Progressive simplification of tetrahedral meshes preserving all isosurface topologies. *Computer Graphics Forum*, 22(3):493–504, 2003.

[CLLR05] Yi-Jen Chiang, Tobias Lenz, Xiang Lu, and Guenter Rote. Simple and optimal output-sensitive construction of contour trees using monotone paths. *Computational Geometry*, 30(2):165 – 195, 2005. Special Issue on the 19th European Workshop on Computational Geometry.

[CM08] Carlos D. Correa and Kwan-Liu Ma. Size-based transfer functions: A new volume exploration technique. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1380–1387, 2008.

[CM09] Carlos D. Correa and Kwan-Liu Ma. Visibility-driven transfer functions. In *Proceedings of the 2009 IEEE Pacific Visualization Symposium*, PACIFICVIS '09, pages 177–184, Washington, DC, USA, 2009. IEEE Computer Society.

[CN94] Timothy J. Cullip and Ulrich Neumann. Accelerating Volume Reconstruction With 3D Texture Hardware. Technical report, University of North Carolina at Chapel Hill, 1994.

[CNSD] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Virtual reality: The design and implementation of the cave. pages 135–142, New York, NY, USA. ACM.

[CNSD+92] Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon, and John C. Hart. The cave: audio visual experience automatic virtual environment. *Commun. ACM*, 35(6):64–72, June 1992.

[CSA03] Hamish Carr, Jack Snoeyink, and Ulrike Axen. Computing contour trees in all dimensions. *Computational Geometry: Theory and Applications*, 24(22):75–94, 2003.

[CSEH07] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete and Computational Geometry*, 37:103–120, 2007.

[CSEM06] David Cohen-Steiner, Herbert Edelsbrunner, and Dmitriy Morozov. Vines and vineyards by updating persistence in linear time. In *Proceedings of the twenty-second annual symposium on Computational geometry*, SCG '06, pages 119–126, New York, NY, USA, 2006. ACM.

[CSvdP04] Hamish Carr, Jack Snoeyink, and Michiel van de Panne. Simplifying flexible isosurfaces using local geometric measures. *IEEE Visualization 2004*, pages 497–504, October 2004.

[CZ09] Gunnar Carlsson and Afra Zomorodian. The theory of multidimensional persistence. *Discrete and Computational Geometry*, 42:71–93, May 2009.

[DCH88] Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '88, pages 65–74, New York, NY, USA, 1988. ACM.

[DFDGTHR04] Remco Duits, Luc Florack, Jan De Graaf, and Bart Ter Haar Romeny. On the axioms of scale space theory. *Journal of Mathematical Imaging and Vision*, 20:267–298, May 2004.

[DFFP03] Remco Duits, Michael Felsberg, Luc Florack, and Bram Platel. α Scale Spaces on a Bounded Domain. *Lecture Notes in Computer Science*, 2695:494–510, 2003.

[dSMVJ10] Vin de Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson. Dualities in persistent (co)homology. Technical report, 2010.

[DW05] C. Dong and G. Wang. Curvatures estimation on triangular mesh. *Journal of Zhejiang University SCIENCE*, 6A:128–136, 2005.

[EH10] Herbert Edelsbrunner and John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010.

[EHNP03] Herbert Edelsbrunner, John Harer, Vijay Natarajan, and Valerio Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, SCG '03, pages 361–370, New York, NY, USA, 2003. ACM.

[EHZ03] Herbert Edelsbrunner, John Harer, and Afra Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete and Computational Geometry*, 30(1):87–107, 2003.

[EKE01] K. Engel, M. Kraus, and T. Ertl. High-Quality Pre-integrated Volume Rendering using Hardware-accelerated Pixel Shading. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, pages 9–16. ACM New York, NY, USA, 2001.

[ELZ02] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete and Computational Geometry*, 28(4):511–533, 2002.

[Far86] Gerald Farin. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3(2):83–127, August 1986.

[Far93] Gerald Farin. *Curves and surfaces for computer aided geometric design (3rd ed.): a practical guide*. Academic Press Professional, Inc., San Diego, CA, USA, 1993.

[FL99] P. Frosini and C. Landi. Size theory as a topological tool for computer vision. *Pattern Recognition and Image Analysis*, 9:596–603, 1999.

[For98] Robin Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134(1):90–145, 1998.

[For01] Robin Forman. A user's guide to discrete Morse theory. In *Proceedings of the 2001 International Conference on Formal Power Series and Algebraic Combinatorics, A special volume of Advances in Applied Mathematics*, page 48, 2001.

[GBHP08] Attila Gyulassy, Peer-Timo Bremer, Bernd Hamann, and Valerio Pascucci. A practical approach to Morse-Smale complex computation: scalability and generality. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1619–1626, 2008.

[GDN+07] Attila Gyulassy, Mark Duchaineau, Vijay Natarajan, Valerio Pascucci, Eduardo Bringa, Andrew Higginbotham, and Bernd Hamann. Topologically clean distance fields. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1432–1439, 2007.

[GKK+12] Attila Gyulassy, Natallia Kotava, Mark Kim, Charles D. Hansen, Hans Hagen, and Valerio Pascucci. Direct feature visualization using morse-smale complexes. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1549–1562, 2012.

[GNP+05] Attila Gyulassy, Vijay Natarajan, Valerio Pascucci, Peer-Timo Bremer, and Bernd Hamann. Topology-based simplification for feature extraction from 3D scalar fields. In *Proceedings of the Conference on Visualization '05*, VIS '05, pages 535–542. IEEE Computer Society Press, 2005.

[GNP+06] Attila Gyulassy, Vijay Natarajan, Valerio Pascucci, Peer-Timo Bremer, and Bernd Hamann. A topological approach to simplification of three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):474–484, 2006.

[GW01] Igor Guskov and Zoë J. Wood. Topological noise removal. In *Proceedings of Graphics Interface 2001*, GRIN'01, pages 19–26, Toronto, Ont., Canada, 2001. Canadian Information Processing Society.

[Gyu08] Attila Gyulassy. Combinatorial construction of morse-smale complexes for data analysis and visualization, 2008.

[Ham93] Bernd Hamann. Geometric modelling. chapter Curvature approximation for triangulated surfaces, pages 139–153. Springer-Verlag, London, UK, UK, 1993.

[Hat01] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 1 edition, December 2001.

[HGM02] H. Hagen, Y. Guan, and R. Moorhead. Interactive surface interrogation using ipt technology, March 2002.

[HKRs+06] Markus Hadwiger, Joe M. Kniss, Christof Rezk-salama, Daniel Weiskopf, and Klaus Engel. *Real-time Volume Graphics*. A. K. Peters, Ltd., Natick, MA, USA, 2006.

[HM03] Runzhen Huang and Kwan-Liu Ma. Rgvis: Region growing based techniques for volume visualization. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, PG '03, pages 355–, Washington, DC, USA, 2003. IEEE Computer Society.

[Hof90] C. M. Hoffmann. A dimensionality paradigm for surface interrogations. *Computer Aided Geometric Design*, 7(6):517–532, October 1990.

[Hos85] Josef Hoschek. Smoothing of curves and surfaces. *Computer Aided Geometric Design*, 2(1-3):97–105, September 1985.

[HPD91] H. Hagen, H. Pottmann, and A. Divivier. Visualization functions on a surface. *Journal of Visualization and Animation*, 2(2):52–58, 1991.

[HRWH05] Katrin Heitmann, Paul M. Ricker, Michael S. Warren, and Salman Habib. Robustness of cosmological simulations. I. Large-scale structure. *The Astrophysical Journal Supplement Series*, 160(1):28, 2005.

[HSG90] H. Hagen, Th. Schreiber, and E. Gschwind. Methods for surface interrogation. In *Proceedings of the 1st conference on Visualization '90*, VIS '90, pages 187–193, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.

[HSS+05] Markus Hadwiger, Christian Sigg, Henning Scharsach, Khatja Bühler, and Markus Gross. Real-Time Ray-Casting and Advanced Shading of Discrete Isosurfaces. *Computer Graphics Forum*, 24(3):303–312, 2005.

[IGH+09] Robert J. Moorhead II, Yanlin Guan, Hans Hagen, Sven Boẗtger, Natallia Kotava, and Christian Wagner. Illustrative visualization: interrogating triangulated surfaces. *Computing*, 86(2-3):131–150, 2009.

[Iij62] T. Iijima. Basic theory on normalization of a pattern (in case of typical one-dimensional pattern). *Bulletin of Electrical Laboratory*, 26:368–388, 1962.

[KCH00] Wolfgang Kollman, Jaqueline H. Chen, and G. Im Hong. Combined Pdf-Sdf Approach to Partially Premixed Turbulent Combustion. In *Proceedings of the 28th Symposium on Internal Combustion, Edinburgh, Scotland*, 2000.

[KD98] Gordon Kindlmann and James W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of the 1998 IEEE Symposium on Volume Visualization*, VVS '98, pages 79–86, New York, NY, USA, 1998. ACM.

[KHW+09] Aaron Knoll, Younis Hijazi, Rolf Westerteiger, Mathias Schott, Charles Hansen, and Hans Hagen. Volume Ray Casting with Peak Finding and Differential Sampling. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1571–1578, Nov-Dec 2009.

[Kin02] G. Kindlmann. Transfer Functions in Direct Volume Rendering: Design, Interface, Interaction. *Course notes of ACM SIGGRAPH*, 2002.

[KK88] E. Kaufmann and R. Klass. Smoothing surfaces using reflection lines for families of splines. *Computer Aided Design*, 20(6):312–316, July 1988.

[KKH02] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional Transfer Functions for Interactive Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002.

[KKH13] Natallia Kotava, Aaron Knoll, and Hans Hagen. Morse-smale decomposition of multivariate transfer function space for separably-sampled volume rendering. *Computer Aided Geometric Design*, 30(6):549 – 556, 2013. ¡ce:title¿Foundations of Topological Analysis¡/ce:title¿.

[KKM05] Henry King, Kevin Knudson, and Neŝa Mramor. Generating discrete Morse functions from point data. *Experimental Mathematics*, 14(4):435–444, 2005.

[KKS+12] Natallia Kotava, Aaron Knoll, Mathias Schott, Christoph Garth, Xavier Tricoche, Christoph Kessler, Elaine Cohen, Charles D. Hansen, Michael E. Papka, and Hans Hagen. Volume rendering with multidimensional peak finding. In Helwig Hauser, Stephen G. Kobourov, and Huamin Qu, editors, *PacificVis*, pages 161–168. IEEE, 2012.

[KLM98] P. Krsek, G. Lukács, and R. R. Martin. Algorithms for computing curvatures from range data. In *The Mathematics of Surfaces VIII, Information Geometers*, pages 1–16, 1998.

[KLPK01] Ju Y. Kang, Kunwoo Lee, Sang K. Park, and Tae-Wan Kim. Efficient Algorithm for Real-time Generation of Reflection Lines. *KSME International Journal*, 15(2):160–171, February 2001.

[Koe84] Jan Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984.

[KPI+03] J. Kniss, S. Premoze, M. Ikits, A. Lefohn, C. Hansen, and E. Praun. Gaussian transfer functions for multi-field volume visualization. In *Proceedings of IEEE Visualization 2003*, pages 497–504, October 2003.

[Kra05] M. Kraus. Scale-Invariant Volume Rendering. In *Proceedings of IEEE Visualization 2005*, pages 295–302. IEEE, 2005.

[Kra08] M. Kraus. Pre-Integrated Volume Rendering for Multi-Dimensional Transfer Functions. *IEEE/ EG Symposium on Volume and Point-Based Graphics*, pages 1–8, 2008.

[KVH84] James T. Kajiya and Brian P Von Herzen. Ray tracing volume densities. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 165–174, New York, NY, USA, 1984. ACM.

[KW03] Jens Krüger and Rüdiger Westermann. Acceleration Techniques for GPU-based Volume Rendering. In *Proceedings IEEE Visualization*, pages 287–292, 2003.

[KWTM03] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Moller. Curvature-based Transfer Functions for Direct Volume Rendering: Methods and Applications. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, page 67. IEEE Computer Society, 2003.

[Lai95] D.H. Laidlaw. *Geometric Model Extraction from Magnetic Resonance Volume Data*. PhD thesis, California Institute of Technology, 1995.

[Law06] T. Lawson. *Topology: A Geometric Approach*. Oxford graduate texts in mathematics. Oxford University Press, 2006.

[LBM+06] D. Laney, P. T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1053–1060, 2006.

[LC87] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics (Proceedings of ACM SIGGRAPH)*, 21(4):163–169, 1987.

[Lev89] M. S. Levoy. *Display of surfaces from volume data*. PhD thesis, Chapel Hill, NC, USA, 1989.

[LGS99] Joachim Loos, Gunther Greiner, and Hans-Peter Seidel. Modeling of surfaces with fair reflection line pattern. In *Proceedings of the International Conference on Shape Modeling and Applications*, SMI '99, pages 256–, Washington, DC, USA, 1999. IEEE Computer Society.

[Lin94] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, 1994.

[LLRR08] L. Linsen, T.V. Long, P. Rosenthal, and S. Rosswog. Surface Extraction from Multi-field Particle Volume Data using Multi-dimensional Cluster Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1483–1490, 2008.

[LLT04] Thomas Lewiner, Helio Lopes, and Geovan Tavares. Applications of Forman's discrete Morse theory to topology visualization and mesh compression. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):499–508, 2004.

[LP90] L. M. Lifshitz and S. M. Pizer. A multiresolution hierarchical approach to image segmentation based on intensity extrema. *IEEE Transactions on Pattern Anaysis and Machine Intelligence*, 12:529–540, June 1990.

[Mar98] R. R. Martin. Estimation of principle curvatures from range data. *International Journal of Shape Modeling*, 4(3-4):99–109, 1998.

[Mat01] Yukio Matsumoto. *An Introduction to Morse Theory*, volume 208 of *Translations of Mathematical Monographs*. American Mathematical Society, 2001.

[Max70] J. C. Maxwell. On hills and dales. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, 40:421–427, 1870.

[Max95] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[MDSB02] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds, 2002.

[MIA+07] Patrick McCormick, Jeff Inman, James Ahrens, Jamaludin Mohd-Yusof, Greg Roth, and Sharen Cummins. Scout: a data-parallel programming language for graphics processors. *Parallel Computing*, 33(10-11):648–662, 2007.

[Mil63] John Milnor. *Morse Theory*. Princeton University Press, 1st edition, 1963.

[MSR07] Evgeni Magid, Octavian Soldea, and Ehud Rivlin. A comparison of gaussian and mean curvature estimation methods on triangular meshes of range image data. *Computer Vision and Image Understanding*, 107(3):139–159, September 2007.

[MW00] D. S. Meek and D. J. Walton. On surface normal and gaussian curvature approximations given data sampled from a smooth surface. *Computer Aided Geometric Design*, 17(6):521–543, July 2000.

[NBH+07] M. L. Norman, G. L. Bryan, R. Harkness, J. Bordner, D. Reynolds, B. O'Shea, and R. Wagner. Simulating Cosmological Evolution with Enzo. *ArXiv e-prints*, May 2007.

[NJB07] PA Navratil, JL Johnson, and V. Bromm. Visualization of Cosmological Particle-based Datasets. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1712–1718, 2007.

[PCMS09] Valerio Pascucci, Kree Cole-McLaughlin, and Giorgio Scorzelli. The TOPORRERY: computation and presentation of multi-resolution topology. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Mathematics and Visualization, pages 19–40. Springer Berlin Heidelberg, 2009.

[Pet02] Sylvain Petitjean. A survey of methods for recovering quadrics in triangle meshes. *ACM Computer Surveys*, 34(2):211–262, June 2002.

[PM90] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

[PSBM07] Valerio Pascucci, Giorgio Scorzelli, Peer-Timo Bremer, and Ajith Mascarenhas. Robust on-line computation of Reeb graphs: simplicity and speed. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[PSL+98] Steven Parker, Peter Shirley, Yarden Livnat, Charles Hansen, and Peter-Pike Sloan. Interactive Ray Tracing for Isosurface Rendering. In *Proceedings of IEEE Visualization '98*, pages 233–238, October 1998.

[PVGFM95] Eric J. Pauwels, Luc J. Van Gool, Peter Fiddelaers, and Theo Moons. An extended class of scale-invariant and recursive scale space filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17:691–701, July 1995.

[Ran04] S. Rana. *Topological Data Structures for Surfaces: An Introduction to Geographical Information Science.* Wiley, 2004.

[RB05] Anshuman Razdan and Myung Soo Bae. Curvature estimation scheme for triangle meshes using biquadratic Bézier patches. *Computer Aided Design*, 37(14):1481–1491, December 2005.

[Ree46] G. Reeb. Sur les points singuliers d'une forme de pfaff complètement intégrable ou d'une fonction numérique. *Comptes Rendus de L'Académie ses Séances, Paris*, 222:847–849, 1946.

[RGH+10] Jan Reininghaus, David Günther, Ingrid Hotz, Steffen Prohaska, and Hans-Christian Hege. TADD: A computational framework for data analysis using discrete Morse theory. In *Mathematical Software – ICMS 2010*, pages 198–208, 2010.

[RGW+03] Stefan Roettger, Stefan Guthe, Daniel Weiskopf, Thomas Ertl, and Wolfgang Strasser. Smart Hardware-accelerated Volume Rendering. In *Proceedings of the Symposium on Data Visualisation (VISSYM)*, pages 231–238, 2003.

[RKE00] S. Röttger, M. Kraus, and T. Ertl. Hardware-Accelerated Volume and Isosurface Rendering based on Cell-Projection. In *Proceedings of IEEE Visualization '00*, pages 109–116. IEEE Computer Society Press Los Alamitos, CA, USA, 2000.

[RKG+11] Jan Reininghaus, Natallia Kotava, David Guenther, Jens Kasten, Hans Hagen, and Ingrid Hotz. A scale space based persistence measure for critical points in 2d scalar fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2045–2052, 2011.

[RKWH11] Jan Reininghaus, Jens Kasten, Tino Weinkauf, and Ingrid Hotz. Combinatorial feature flow fields: Tracking critical points in discrete scalar fields. Technical Report 11-02, Zuse Institute Berlin, 2011.

[Rob99] Vanessa Robins. Toward computing homology from finite approximations. In *Topology Proceedings*, volume 24, pages 503–532, 1999.

[RSKK06] Christof Rezk Salama, Maik Keller, and Peter Kohlmann. High-level user interfaces for transfer function design with semantics. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1021–1028, 2006.

[RWS11] Vanessa Robins, Peter John Wood, and Adrian P. Sheppard. Theory and algorithms for constructing discrete morse complexes from grayscale digital images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1646–1658, 2011.

[SGG02] G. Sussner, G. Greiner, and R. Grosso. Generating high quality meshes for interactive examination of surface quality on car bodies, 2002.

[Ski89] Steven S. Skiena. Problems in geometric probing. *Algorithmica*, 4(1-4):599–605, 1989.

[Sma61a] S. Smale. Generalized Poincaré's conjecture in dimensions greater than four. *Annals of Mathematics*, 74:391–406, 1961.

[Sma61b] S. Smale. On gradient dynamical systems. *Annals of Mathematics*, 74:199–206, 1961.

[SR04] J.P. Schulze and A. Rice. Real-time volume rendering of four channel data sets. In *Proceedings of IEEE Visualization '04*, pages 598–34. IEEE Computer Society, 2004.

[Sra94] M. Sramek. Fast Surface Rendering from Raster Data by Voxel Traversal Using Chessboard Distance. *Proceedings of IEEE Visualization 1994*, pages 188–195, 1994.

[SSWB05] Kurt Stockinger, John Shalf, Kesheng Wu, and E. Wes Bethel. Query-driven visualization of large data sets. In *Proceedings of the Conference on Visualization '05*, VIS '05, pages 167–174. IEEE Computer Society, 2005.

[Tau95] G. Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proceedings of the Fifth International Conference on Computer Vision*, ICCV '95, pages 902–, Washington, DC, USA, 1995. IEEE Computer Society.

[TF97] Holger Theisel and Gerald Farin. The curvature of characteristic curves on surfaces. *IEEE Computer Graphics and Applications*, 17(6):88–96, November 1997.

[TGK+04] Xavier Tricoche, Christoph Garth, Gordon Kindlmann, Eduard Deines, Gerik Scheuermann, Markus Ruetten, and Charles Hansen. Visualization of intricate flow structures for vortex breakdown analysis. *Proceedings of IEEE Visualization '04, 2004*, pages 187–194, 2004.

[TGSP09] Julien Tierny, Attila Gyulassy, Eddie Simon, and Valerio Pascucci. Loop surgery for volumetric meshes: Reeb graphs reduced to contour trees. *IEEE Transactions on Visualization and Computer Graphics*, 15:1177–1184, 2009.

[TLM03] Fan-Yin Tzeng, Eric B. Lum, and Kwan-Liu Ma. A novel interface for higher-dimensional classification of volume data. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 505–512, Washington, DC, USA, 2003. IEEE Computer Society.

[TLM05] Fan-Yin Tzeng, Eric B. Lum, and Kwan-Liu Ma. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):273–284, 2005.

[TNTF04] Shigeo Takahashi, Gregory M. Nielson, Yuriko Takeshima, and Issei Fujishiro. Topological volume skeletonization using adaptive tetrahedralization. In *Proceedings of the Geometric Modeling and Processing 2004*, GMP '04, pages 227–, Washington, DC, USA, 2004. IEEE Computer Society.

[VOE97] C. W. A. M. Van Overveld and B. Wyvill TU Eindhoven. An algorithm for polygon subdivision based on vertex normals. In *Proceedings of the 1997 Conference on Computer Graphics International*, CGI '97, pages 3–12, Washington, DC, USA, 1997. IEEE Computer Society.

[VPBM01] Alex Vlachos, Jörg Peters, Chas Boyd, and Jason L. Mitchell. Curved pn triangles. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, I3D '01, pages 159–166, New York, NY, USA, 2001. ACM.

[WB01] Kouki Watanabe and Alexander G. Belyaev. Detection of salient curvature features on polygonal surfaces. *Computer Graphics Forum*, 20(3):385–392, 2001.

[WDC+07] Gunther H. Weber, Scott E. Dillard, Hamish Carr, Valerio Pascucci, and Bernd Hamann. Topology-controlled volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13:330–341, March 2007.

[Wes90] Lee Westover. Footprint Evaluation for Volume Rendering. In *SIGGRAPH '90: Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, pages 367–376, New York, NY, USA, 1990. ACM Press.

[WHDS04] Z. Wood, Hugues Hoppe, M. Desbrun, and P. Schröder. Removing excess topology from isosurfaces. *ACM Transactions on Graphics*, 23(2):190–208, 2004.

[WII99] Joachim Weickert, Seiji Ishikawa, and Atsushi Imiya. Linear scale-space has first been proposed in Japan. *Journal of Mathematical Imaging and Vision*, 10:237–252, May 1999.

[Wit83] Andrew P. Witkin. Scale-Space Filtering. In *8th International Joint Conference on Artificial Intelligence*, volume 2, pages 1019–1022, Karlsruhe, August 1983.

[Wu91] X. Wu. An Efficient Antialiasing Technique. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, pages 143–152. ACM, 1991.

[Zom01] Afra Zomorodian. *Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes*. PhD thesis, Urbana, Illinois, October 2001.

Curriculum Vitae

12.2008 - 02.2012	PhD Student in Computer Graphics Group, Department of Computer Science, University of Kaiserslautern, Germany
10.2005 - 11.2008	Master Student at the Department of Computer Science, Majoring in Computer Graphics, University of Kaiserslautern, Germany
10.2003 - 09.2005	Master Student at the Department of Mathematics, Majoring in Industrial Mathematics, University of Kaiserslautern, Germany
09.1998 - 07.2003	Student at the Department of Computer Science, Belarussian State University of Informatics and Radioelectronics, Minsk, Belarus

List of Publications

[1] Natallia Kotava, Aaron Knoll, and Hans Hagen. *Morse-smale decomposition of multivariate transfer function space for separably-sampled volume rendering.* Computer Aided Geometric Design, 30(6):549 – 556, 2013.

[2] Natallia Kotava, Aaron Knoll, Mathias Schott, Christoph Garth, Xavier Tricoche, Christoph Kessler, Elaine Cohen, Charles D. Hansen, Michael E. Papka, and Hans Hagen. *Volume rendering with multidimensional peak finding.* In Helwig Hauser, Stephen G. Kobourov, and Huamin Qu, editors, PacificVis, pages 161–168. IEEE, 2012.

[3] Attila Gyulassy, Natallia Kotava, Mark Kim, Charles D. Hansen, Hans Hagen, and Valerio Pascucci. *Direct feature visualization using morsesmale complexes.* IEEE Transactions on Visualization and Computer Graphics, 18(9):1549–1562, 2012.

[4] Jan Reininghaus, Natallia Kotava, David Guenther, Jens Kasten, Hans Hagen, and Ingrid Hotz. *A Scale Space Based Persistence Measure for Critical Points in 2D Scalar Fields.* IEEE Transactions on Visualization and Computer Graphics, 17(12):2045–2052, 2011.

[5] Robert J. Moorhead, Yanlin Guan, Hans Hagen, Sven Böttger, Natallia Kotava, and Christian Wagner. *Illustrative visualization: interrogating triangulated surfaces.* Computing, 86(2-3):131–150, 2009.